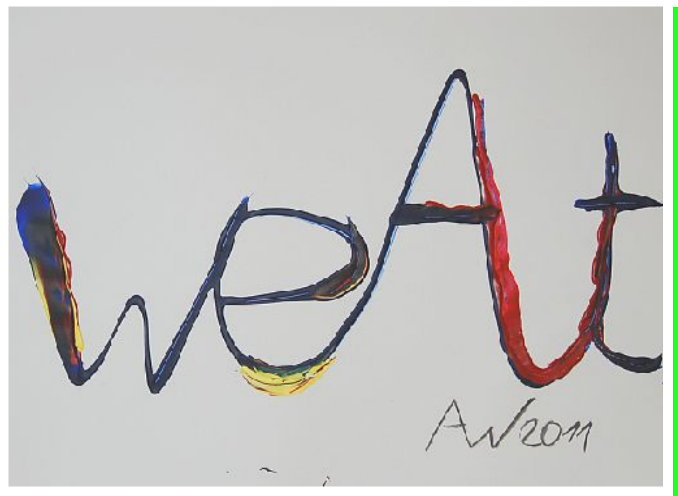
Albrecht Weinert
weinert - automation

weAut_01 automation controller

Benutzerhandbuch





Stand: 09.05.2013

weinert – automation   Bochum
Prof. Dr.-Ing. Albrecht Weinert

Schattbachstraße 42

D-44801 Bochum

Telefon: +49 (0) 234 7089634

Telefax: +49 (0) 3212 12338681

E-Mail: info@weinert-automation.de

Internet: weinert-automation.de

Steuernummer: 350/5264/0534 Finanzamt Bochum Süd

Umsatzsteuer-Identifikationsnummer gemäß § 27 a UStG: DE277642087

weAut_01 — Benutzerhandbuch (de 01)

Versionshistorie:

V.01.00, 01.08.2011: Neu (aus Entwicklungsunterlage V.02.00)

V.01.04, 29.09.2011: Ergänzungen, Inbetriebnahme ausführlicher

V.01.05, 23.11.2011: Ethernet, ISP, div. Ergänzungen

V.01.07, 20.12.2012: einige Aktualisierungen

Version: V.01.07

Zuletzt geändert von A. Weinert am 09.05.2013

Versionsverwaltung (SVN):
https://ai2t.de/svn/indUcDoc/pub/weAut_01/weAut_01-UserMan-de01.odt

Download: http://weinert-automation.de/files/weAut_01/weAut_01-UserMan-de01.pdf

Hinweis: Tabellen, Listings, Bilder etc. sind gemeinsam durchnummeriert.

Inhalt

1. Einleitung

weAut_01 ist eine µController-basierte, leistungsfähige Automatisierungsbaugruppe mit

- 16 Kanälen Prozess-Ein/Ausgabe nach Industriestandard,
- 2 Kommunikationsschnittstellen, nämlich RS232 und Ethernet,
- und einigem Anderem mehr.

Mit dieser Baugruppe können Sie verteilte, vernetzte oder isolierte (stand alone) Automatisierungsaufgaben flexibel lösen. Neben dem internen EEPROM kann auch eine SMC (small memory card) für nicht flüchtige (persistente) Speicherung eingesetzt werden.

1.1 Inhalt und Zielgruppe (intended audience)

Dies Handbuch ist die Beschreibung der hardware- und softwaretechnischen Handhabung der Baugruppe weAut_01 von weinert – automation. Es liefert die Informationen für

- die Anbindung an die Prozesssignale, und an Kommunikationsschnittstellen
- die Stromversorgung (Lastspannung, redundante Speisung) der Baugruppe
- optionale Erweiterung und den Einbau in Gehäuse
- das Schreiben der Software und die Programmierung des µControllers.

Die Software kann zum einen von Null an (from scratch) erstellt werden. Zum Anderen kann sie auf Basis des leistungsfähigen Laufzeitsystems weAutSys geschrieben werden. Dieses System stellt neben Hardwaretreibern und zahlreichen Hilfsfunktionen (utilities) eine PLC-ähnliche Umgebung mit periodischen Zyklen, Timern etc. zur Verfügung. Des Weiteren integriert sie einen bootloader, der jegliche zusätzliche Programmierhardware entbehrlich macht.

Das erste Vorgehen "software from scratch" ist i.A. nur für extrem einfache oder sehr komplizierte / spezielle Fälle angebracht. Die überwiegende Anzahl der Anwendungen wird effektiver auf Basis von weAutSys erstellt.

Die Handbuch wendet sich also vorwiegend an den technisch versierten Anwender bzw. Anwendungsentwickler beim Kunden. Einen kurzen Überblick über die Eigenschaften im Sinne eines "executive summary" geben die Kapitel 2.1 und 2.2.

1.2 Copyright

Das vorliegende Handbuch, Schaltung und Layout der Baugruppe, wesentliche Teile der Software weAutSys, sowie die meisten Bilder und Abbildungen sind, mit Copyright Albrecht Weinert, urheberrechtlich geschützt.

Die betreffenden Produkte, Informationen und Dateien dürfen nur von Kunden im Rahmen der von weinert – automation erworbenen Lizenz verwendet werden. Sie dürfen sie ohne ausdrückliche Genehmigung nicht weitergeben.

Ein Teil der Informationen stammt erkennbar von den Herstellern der elektronischen Bausteine. Ein Teil der Laufzeitsoftware ist angepasste und verbesserte open source software. Diese Anteile stehen unter LPGL oder BSD Lizenz. Sie sind somit, anders als z.B. GPL auch für kommerzielle Produkte / Systeme verwendbar.
Auch diese Informationen und Werke von Dritten sind urheberrechtlich geschützt. Respektieren Sie bitte auch deren Lizenzbedingungen und Rechte.

2. Eigenschaften — Daten

2.1 Steckbrief

Die Automatisierungsbaugruppe weAut_01 kombiniert

- den Einsatzes preiswerter Standard - µController
- mit geringem Stromverbrauch im Sinne "green automation" und
- die Nutzung geeigneter open source - Ansätze sowie
- die Anwendungsprogrammierung in einer höheren Sprache (C)

mit industriegerechten Konzepten bei

- Prozessperipherieschnittstellen
- Stromversorgung
- Überwachung
- Schutz

Der Ansatz platziert sich bewusst zwischen

A) PC- (Zusatz-) Platinen und
µControllerboards einerseits und

B) kompakter industrieller Automatisierungstechnik andererseits.

Die Genügsamkeit bei Kosten, Strom- und Ressourcenverbrauch wird erreicht mit:

- Ressourcen sparender Betriebssoftware
- Bei Verwendung von Fremd-Software nur
  a) lizenzgebührenfreie und mit
  b) für kommerziellen Einsatz geeigneter Lizenz.
- Verwendung nur von Standard-Kommunikationsschnittstellen
  wie V.24 / RS232 und
  Ethernet mit allen TCP-/UDP-IP-Möglichkeiten
- Verzicht auf ein vorgegebenes, festes Aufbaukonzept

weAut_01 ist vom Hard- und Softwarekonzept her ein geschlossenes, fertiges Modul. Es ist zwar durch Bestückungsvarianten und mit ggf. Zusatzmodulen modifizierbar, aber dies wird aus Aufwandgründen nicht direkt unterstützt.

Erweiterungen über die gegebenen I/O-Möglichkeiten hinaus sollen vom Ansatz her durch (Ethernet-) Vernetzung implementiert werden.

Die Baugruppe lässt sich mit einer lizenzgebührenfreien Werkzeugkette auf gut dokumentierte Weise mit allen Möglichkeiten und Freiheiten in C programmieren.
Eine PLC-/SPS-Gewohnheiten entsprechende Basis von Zyklen, Timern und zahlreichen Grunddiensten bietet optional das Laufzeitsystem weAutSys.

Die qualitativen Vorzüge im Sinne von industrieller Automatisierung / Prozesssteuerung ergeben sich durch:

- industriegängige Schnittstellen
- industriegerechte Spannungsversorgung
- professionelle Schutzkonzepte für alle Schnittstellen
- Möglichkeit industriegerechter Aufbau- und Gehäusevarianten vereinfacht durch einseitige Bestückung
- hohe Qualität der Softwareerstellung bzw. der -auswahl.

Anmerkung: Diese unvollständigen Aufzählung von Qualitätsvorzügen gegenüber Produkten der Kategorie A) bezieht sich auf den Einsatz in prozessnaher Automatisierung. Sie soll hochwertige Produkte dieser Kategorie keineswegs abwerten. Diese sind von ihrem Design her aber kaum für den direkten Anschluss an Prozessperipherie bzw. an Feldgeräte, für längere Anschlussleitungen, höheren Spannungsniveaus und EMI-Anforderungen konzipiert. Wer solche Produkte, insbesondere wenn sie Controller-Anschlüsse direkt an Klemmen herausführen dennoch so einsetzt, nimmt massive Zuverlässigkeits- und oft auch Sicherheitsmängel in Kauf.

2.2 Technische Daten

Leiterplatte: 151 * 82 mm; zweilagig; einseitig bestückt; überwiegend SMD.

Prozessor: AVR ATMega1284

Speicher: 128K Flash (Programmspeicher), 16K RAM,
4K EEPROM (nicht flüchtig)

small memory cards (SMC, via SPI) einsetzbar (nicht flüchtig)

Ethernet: 10BaseT; 28J60; RJ45.

V.24/RS232: SubD 9pol.; +/- 12V; MAX202

Programmierung / Erweiterung: ISP (10); 2Wire / I²C.

Prozess-I/O: 8 DI 24V / 12 V (190V max.) auch verwendbar als
8 AI 8 /10 Bit; 4V, 10V, 18V + Overflow
8 DO Lastspannung; max. 100mA; kurzschlussfest
Zähler / Generator (auch via SubD und MAX202)

Lastspannung / Versorgung: 24V / 12V (+10..+30V max.)

redundante Einspeisung: 24V / 12V (10..30V max.) neben der Lastspannung.
versorgt redundant / zusätzlich alles außer DO / Lastspannung;
ist geeignet auf für Pufferbatterie (9..24V)
und Wechselstromspeisung (AC, 18V~ max.).

Erdung: brückbare Trennung zwischen Schutzerde (PE Gehäuse, Schirme)
und Signalerde / Lastspannung „M“ (Gnd)

Schutz: jeweils geeignete Schutzbeschaltungen
für alle Versorgungs- und I/O-Klemmen.
(Sicherung, Schutzdioden und -widerstände, robuste Treiber etc.)

Überwachung der Lastspannung (per Komparator und Software).
zusätzlich softwareunabhängige Anzeige
"Lastspannung vorhanden" bzw.
"redundante Einspeisung vorhanden" durch jeweils zwei blaue LEDs.

2.3 Aufbau, Leiterplatte, Gehäuse

Die Automatisierungsbaugruppe weAut_01 hat eine zweilagige starre Leiterplatte; sie ist einseitig und vorwiegend mit SMD-Bauteilen bestückt. Die einseitige Bestückung bietet neben einigen Kosten-, Handhabungs- und Montagevorteilen die EMV-Vorzüge (fast) durchgängiger Ground- bzw. Vcc-Planes, die zweilagig sonst nicht zu haben wären.

Eine Aufbautechnik wird dem Kunden bei der Baugruppe weAut_01 nicht vorgegeben. Wahl und Einsatz eines geeigneten und ggf. notwendigen Gehäuses liegt in Anwender-verantwortung.

Bild 1 zeigt die Platinenabmessungen. Die Platine passt in ein entsprechend breites Hutschienengehäuse. Passend bei Abmessungen und Schraubbohrungen ist beispielsweise das 157 mm breite Gehäuse "Wöhr SERIE 1570"; vgl. Bild 2 ([Wöhr 1050_deu.pdf, 1570_deu.pdf]). Diese Wöhr-Gehäuse lassen sich mit zwei Einstecklaschen auch ohne Hutschiene an eine Wand schrauben. Taster und LEDs lassen sich mit Betätigungsachsen und Lichtleitern (Zubehör) an die Frontplatte führen.

Alternativ zum Hutschienengehäuse lässt sich die Baugruppe auch in einem (serienmäßigen) Flachgehäuse mit Kabeldurchführungen einbauen, das dann auch die Anschlüsse komplett verdeckt.

Weitere Gehäuse (Kunststoff, Aluminium) sind bei entsprechender Stückzahl lieferbar.

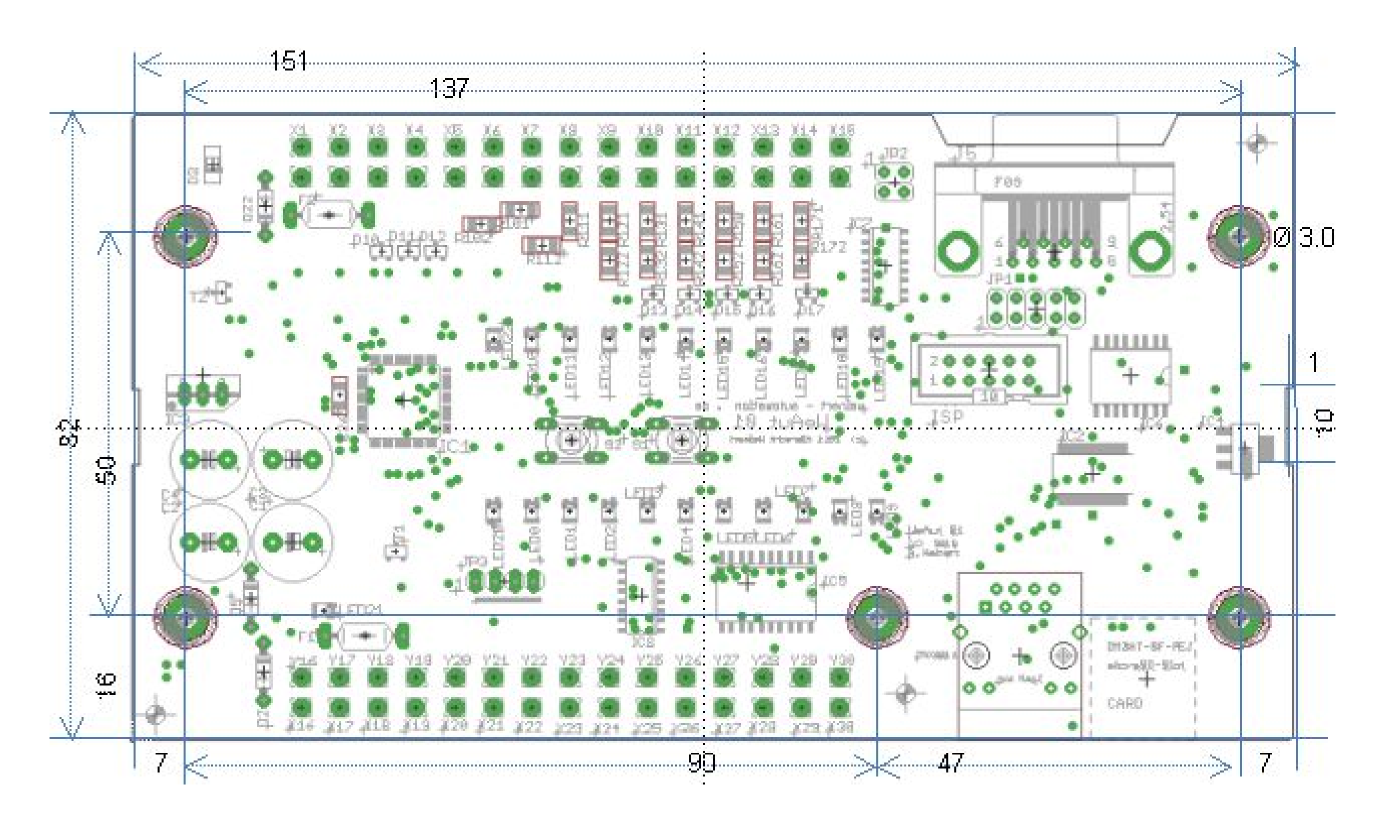


Bild 1: Platinenabmessungen

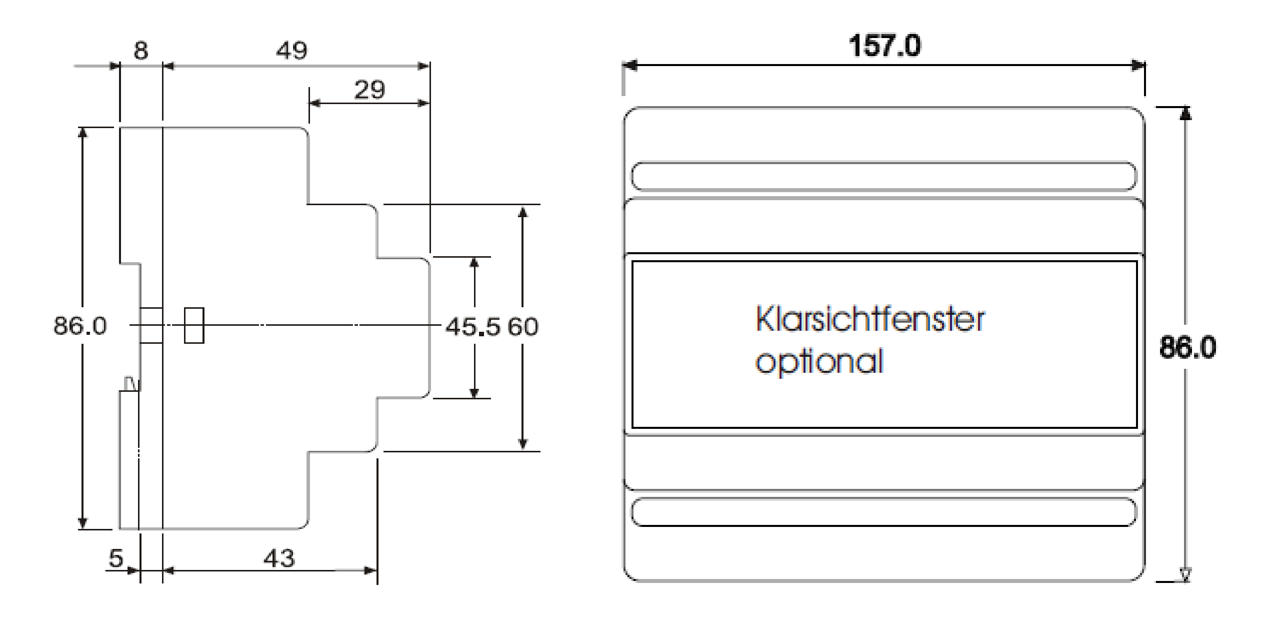


Bild 2: Profil und Front eines passenden Hutschienengehäuses [Quelle Wöhr]

2.4 Grundstruktur / Funktionsblöcke

Bild 3 zeigt die Grundstruktur von weAut_01.

- Die grün gehaltenen Komponenten sind Teil des µControllers Atmega1284P.
- Rot dargestellte Systemteile vermitteln die (Prozess-) Schnittstellen nach außen.

Alle Komponenten werden in den zugehörigen Kapiteln erläutert.

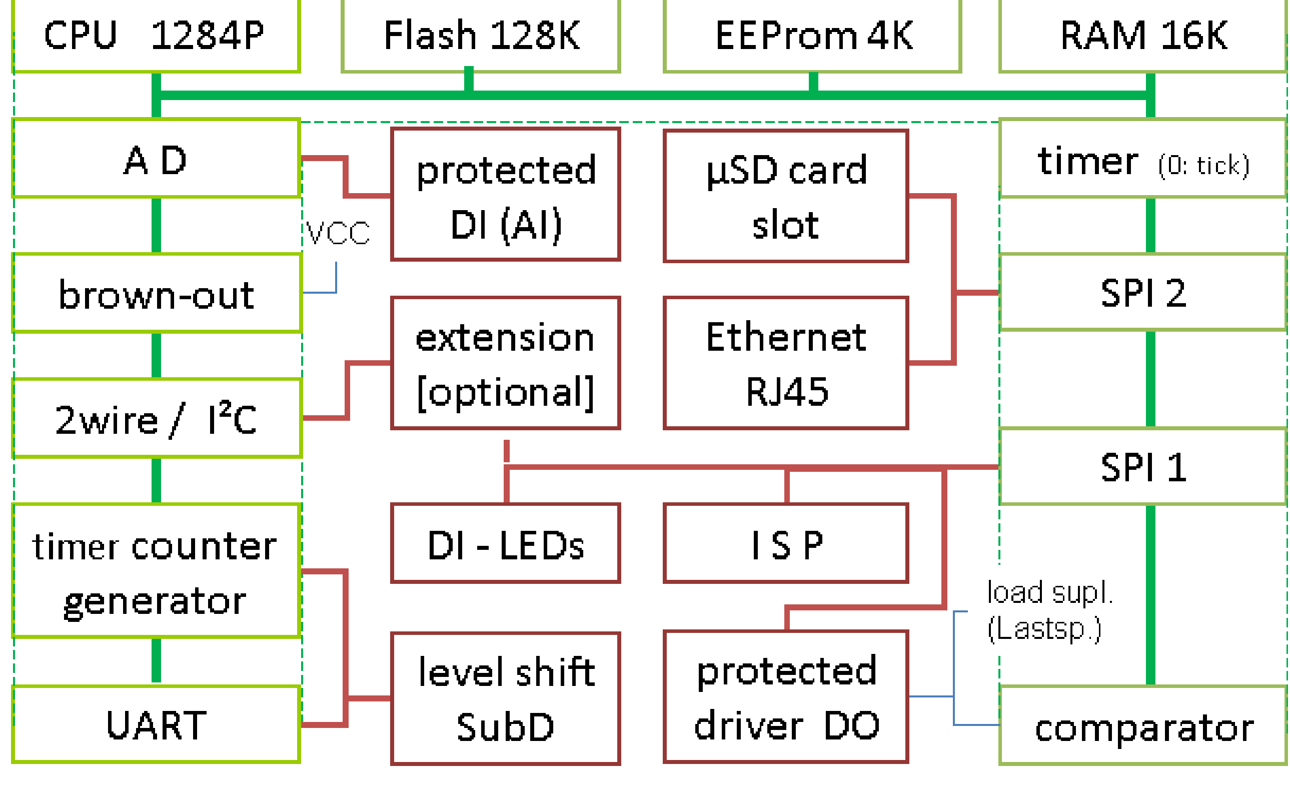


Bild 3: Blockbild CPU: ATmega1284P; Ethernet: Enc28j60

Prozessor

Die eingesetzte CPU ATmega1228P hat 128 KByte Flash (Programmspeicher), 4KByte EEPROM und 16KByte RAM (Arbeitsspeicher). Der Prozessor wird 20 MHz und 5V betrieben. Dies ergibt seine maximal mögliche Arbeitsgeschwindigkeit.

Hinweis: Für weitgehend verminderte Anforderungen, insbesondere bezüglich der Ethernet-Kommunikation, ist auch eine Bestückungsvariante mit der CPU Atmega644P denkbar (Flash: 64K; EEPROM:2K, RAM 4K).

2.3 Anwendung

Die Baugruppe ist ausgestattet mit:

- gängigen Prozessschnittstellen,
- ein effektives Laufzeitsystem weAutSys
  mit PLC ähnlichen Zyklen (threads)
  und Möglichkeiten, wie u.a. Timer,
- die Anwendungsprogrammierung in C,
- die flexible Speisung sowie auch
- der geringe Stromverbrauch
- direkte Ansteuermöglichkeit von Relais und Schützen (12 / 24V)
- direkte Ansteuerung von LED-Leuchten (12 / 24V)
- direkte Erfassung analoger und binärer Prozesssignale

Diese Merkmale eröffnen vielfältige Anwendungsmöglichkeiten:

- Ausgelagerte dezentrale Peripherie
- Zähler, Energiemanagement.
- Hausautomation (12V)
- industrielle Automation (24V)
- I/O-Konzentrator (und Vorverarbeiter) für beispielsweise PC-basierte Automatisierung via Standard Ethernet Kopplung
- Robuste Basis-Baugruppe für die µController - Ausbildung
- Steuerung oder Automatisierung von Modellen und Laboraufbauten

Die hier nur angedeutete Breite der möglichen Anwendungsszenarien kann durch Bestückungsvarianten und oder Varianten des Laufzeitsystems noch erweitert werden.

3. Die Komponenten

3.1 Bedienelemente und Anschlüsse

Bild 4 zeigt die Lage der Bedien- und Anzeigeelemente. Absolute Maße beziehen sich auf die linke untere Ecke der Leiterplatte.

Die SMD-LEDs können mit Lichtwellenleitern auch an weiter entfernten Frontplatten sichtbar gemacht werden. Desgleichen lassen sich die Taster S2 „enter“ und S1 „reset“ mit entsprechenden Kunststoffstangen (z.B. Potiachsen) von entfernteren Frontplatten aus betätigen. Die Betätigung von „reset“ sollte mit der Frontplatte bündig abschließen bzw. gegen versehentliches Betätigen sogar etwas versenkt sein.

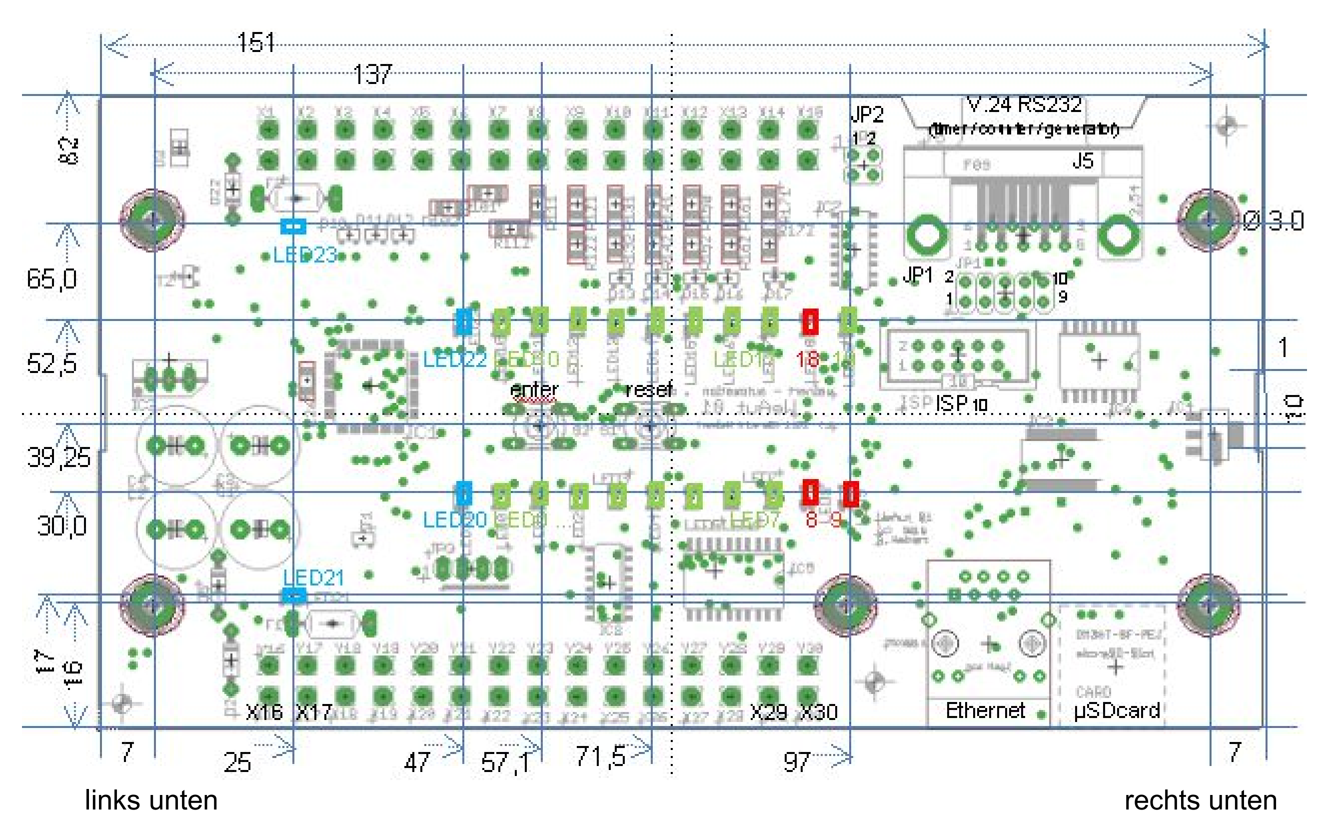


Bild 4: Lage der B&B- und Anschlusselemente

Anschlussklemmen

Tabelle 5 zeigt die Anschlussklemmen die jeweils am oberen und unteren Rand der Baugruppe angeordnet sind. Es können sowohl Schraubklemmen (äußere Lötaugenreihe) als auch Steckverbinder, auch für steckbare Schraubklemmen, (innere Lötaugenreihe) bestückt werden. Das Rastermaß für die Klemmen ist 5 mm.

X1 und X16 sind PE (Schutzerde), welche vom gemeinsamen Bezugspotential Gnd getrennt ist. Mit PE sind alle 5 Befestigungsbohrungen und die Gehäuse von Ethernet- und der V.24-Buchse sowie des SMCard-Halters verbunden. Ein Widerstand zwischen PE und Gnd (30K, 1/4W) verhindert ein „Weglaufen“ im Falle einer isolierten Versorgung.

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15
oben	PE	Gnd	L+ via Dio & R	redund. Einspeis.	Gnd	Gnd	DI0 AI0	DI1 AI1	DI2 AI2	DI3 AI3	DI4 AI4	DI5 AI5	DI6 AI6	DI7 AI7	rangierbar JP2
unten	PE	Gnd	Lastspannung	Lastspannung	Gnd	Gnd	DO0	DO1	DO2	DO3	DO4	DO5	DO6	DO7	4K64-Gnd
	X16	**X17**	**X18**	**X19**	**X20**	**X21**	**X22**	**X23**	**X24**	**X25**	**X26**	**X27**	**X28**	**X29**	**X30**

Tabelle 5: Anschlussklemmen oben (neben SubD) und unten (neben RJ45 und SMC)

Die Lastspannung (X18 und X19) versorgt die digitalen Ausgänge sowie die übrige Elektronik, während die redundante Einspeisung (X4) ggf. nur letztere speist. Die höhere Spannung beider Einspeisungen steht über eine Diode und einen Widerstand 2K5 an X3 zur Verfügung, z.B. als kurzschlussfeste Versorgung der Eingänge DI0 .. DI7.

X30 bietet einen Widerstand (4K64, 1%, max. 30V) gegen Gnd, der z.B. als Last für einen Ausgang (DOx) oder einen Kontakt dienen kann.

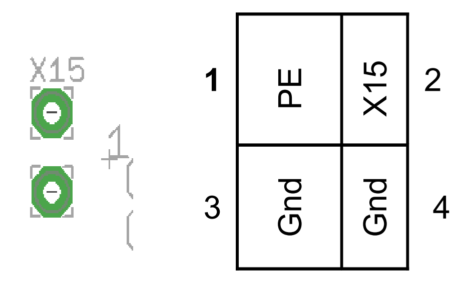


Bild 6: JP2 neben X15 zur beliebigen Verwendung von X15 oder als Gnd oder PE

Anzeigeelemente

Zwei LEDs gn und ge im RJ45 Stecker (Ethernet)
auf parametrierbare Weise angesteuert durch Ethernet-Controller (ENC28J60; z.B. gn: Link status ge: transmit), d.h. also zur flexiblen Anzeige von Schnittstellenzuständen oder auch frei verwendbar

Zwei LEDs bl: für „Lastspannung da“ (> ca.10V)

Zwei LEDs bl: je für „redundante Einspeisung vorhanden“(> ca. 10V).

Acht LEDs gn: für die binären / digitalen Ausgänge

Eine LED rt: für Fehler bei mindestens einem digitalen Ausgang (Überstrom oder thermische Abschaltung)

Eine LED gn: für Freigabe (enable) der digitalen Ausgänge

Acht LEDs gn: für die digitalen Eingänge.
(hierfür gedacht, aber prinzipiell frei verwendbar.)

Eine LED rt: freie Verwendung (PB4) durch die Software (Lastspannung weg)

Eine LED gn: freie Verwendung (PB2) durch die Software (Run)

Bedienelemente

eine Reset-Taste

eine Taste mit freier Verwendungsmöglichkeit (PB0, Arbeitsname "enter")

Element	Funktion	Hinweis / Anmerkung
S1 „enter“	Port B0 auf Lo	von der Software auszuwerten
S2 „reset“	Reset; unmaskierbar	wirkt direkt auf den µController
LED 0 gn	DO 0 Hi	direkt mit der Ansteuerung des betreffenden DO-Ausgangstreibers gekoppelt
LED 1..7 gn	DO 2..7 Hi	
LED 8 rt	DO /fault (=PC4)	vom DO-Treiber-IC (Lo bei Überlast)
LED 9 rt	DO disable (= PC5)	wenn Lo = LED An sind DO0..7 aus
LED 10 gn	DI 0 (für Zustand DI)	von der SW frei ansteuerbar; der Zusammenhang mit DI ist gewollt, nicht erzwungen.
LED 11..17	DI 1 ..7 gn dto.	
LED 18 rt	Port B 4 Lastsp. weg	+ freie Verwendung durch die SW (Test-LED)
LED 19 gn	Port B 2 “Run“	+ freie Verwendung durch die SW (Test-LED)
LED 20 bl	Lastspannung vorhanden	leuchtet ab etwa 12V mit konstanter Helligkeit
LED 21 bl		wie LED 20
LED 22 bl	redundante Einspeisung vorhanden	leuchtet ab etwa 12V mit konstanter Helligkeit
LED 23 bl		wie LED 22

Tabelle 7: Anzeige- und Bedienelemente

Tabelle 7 zeigt die Funktionen der Anzeige-LEDs und der Bedientasten. Der dort dargestellte Zusammenhang zwischen dem Zustand der auch als Analogeingabe verwendbaren digitalen Eingänge DI 0 .. DI 7 und den grünen LED 10 .. LED 17 ist so gewollt und von der Anordnung der Klemmen und der LEDs (vgl. Bild 4) *die* sinnvolle Verwendungsmöglichkeit dieser LEDs. Dieser Zusammenhang wird allerdings durch die Software, die die Eingänge bearbeitet und ggf. filtert, hergestellt. Insofern ist auch eine beliebige andere Verwendung der LED 10 .. LED 17 möglich.

Hinweis: Bei als AE verwendeten Eingängen wäre ein sinnvoller Zusammenhang zwischen Eingangsspannung und Anzeige "grün" herzustellen. Dies ist aber nur anwendungsspezifisch darstellbar.

Im Gegensatz dazu ist der Zusammenhang der grünen LED 0 .. LED 7 und der Ansteuerung der digitalen Ausgänge DO 0 .. DO 7 per Hardware fest vorgegeben. Diskrepanzen zwischen LED 0..7 und DO 0..7 in dem Sinne, dass eine LED leuchtet und der zugehörige Ausgang Lo ist, ergeben sich nur durch die in Tabelle 8 dargestellten Umstände.

LED 9 DOdisab PC5	**LED8 DOfault PC4**	**LED 20/21 Lastspannung. da**	**LED 18 Lastsp. weg PB4**	**Zustand DO 0.. DO 7 verglichen mit LED-Anzeigen LED 0 .. LED 7** (X = don't care x = 0..7)
Aus	Aus	An	Aus	Übereinstimmung: LEDx an = DOx Hi/on
An	X	X	X	Alle DOx Lo/off (unabhängig von LEDx)
X	X	Aus *1)	An*1)	Kein DOx mit brauchbarer Ausgangsspannung
Aus	An *2)	X	X	Mindestens ein DOx ist Lo trotz LEDx an

Tabelle 8: Anzeige der LED 0 .. LED 7 und Zustand der Ausgänge DO 0 .. DO 7

Hinweis zu *1): Diesen Zustand mangelnder Lastspannung (<10V bzw. <19V) sollte die Software mit DO disable eindeutig machen.
*2) Der Ausgangstreiber schaltet bei Überlast / Übertemperatur u.U. alle Kanäle ab. Das Rücksetzen geschieht per SW mit kurzzeitigem DO disable.

Die kurz gefasste Interpretation von Tabelle 8 ist:
Wenn keine der drei roten LEDs 8, 9 und 18 leuchten, sind die digitalen Ausgänge OK.
Ihr Ausgangszustand entspricht der jeweiligen Anzeige der LED0..LED7.

3.2 Stromversorgung

Die Gleichspannungsversorgung der digitalen Prozess-Ausgabe und der Sensoren der Prozesseingabe wird üblicherweise auch „Lastspannung“ (load voltage) genannt.

- In der industriellen Automatisierungstechnik ist diese üblicherweise 24V mit einem Toleranzbereich von etwa 19 .. 27V.
- Bei der Gebäudetechnik (Steuerung, Schutzsysteme, Alarmanlagen) und in der (Nicht-Nutz-) Fahrzeugtechnik ist die Lastspannung überwiegend 12V (10..14V).

weAut_01 überdeckt beide Bereiche.

Lastspannung nominell 12V oder 24V

Stromaufnahme ohne Prozesslast : 200, 90 .. 40 mA (bei Anlauf, 12V .. 24V);
siehe Hinweise 1 und 2 unten.

Stromaufnahme durch Prozesslast:: 0..900mA
je nach Ansteuerung und Belastung der digitalen Ausgänge

Erlaubter Bereich: 10 .. 28V

Absolute max. rating: -0,5V... +30V; kurzzeitig (0,1ms) 35V

Redundante Einspeisung: nominell 12V oder 24V

Stromaufnahme: 200, 90 .. 40 mA (bei Anlauf, 12 .. 24V)

Hinweis 2: Diese Stromaufnahme trägt entweder die Lastspannung oder die redundante Einspeisung, letztere falls sie etwas (500mV) höher ist als die Lastspannung.

Erlaubter Bereich: 10 .. 29V (120 ...60 mA,; vgl. Bild 9)

Absolute max. rating: -50V...+30V; kurzzeitig (0,1ms) 35V
(Wechselspannungsspeisung möglich, s.u.)

Die redundante Einspeisung kann unter anderem auch

- mit der Lastspannung verbunden werden, und sie sollte es, wenn keine weitere Versorgung zur Verfügung steht, da dann alle Schutz- und Überspannungsdioden im Einsatz sind.
- mit externen Pufferkondensatoren verbunden werden, die über Dioden von der Lastspannungsversorgung geladen werden. Damit lassen sich die Pufferzeiten für den Prozessor, Eingaben etc. erhöhen.
- mit einem (gepufferten) Akku oder einer Batterie 9,6 .. 24V verbunden werden
- an eine Wechselspannung 8..12V_{eff} angeschlossen werden (Hutschienentrafo z.B.)

Überwachung der Einspeisungen:

- zwei blaue LEDs für Lastspannung; gut sichtbare Helligkeit ab etwa 11V
- zwei blaue LEDs für die redundante Einspeisung;
  gut sichtbare Helligkeit ab etwa 11V; vgl. Bild 9.
- Komparator (µController - Port - Funktion) mit Schwelle bei etwa 10,5V *1)
  nur für die Lastspannung,
  vorgesehene Anzeige des Ausfalls per Software über die rote LED18.

Die Anzeigen der (2*2) blauen LEDs sind völlig unabhängig vom µController und dessen Software und funktionieren auch während Reset oder ISP.

Die Komparator-Funktion ist hingegen per Software für den PB3 zu erbringen. Die mit einer Diode entkoppelte Lastspannung wird geteilt (37,4K \ 4K64) mit der internen Bandgap-Referenz (1,1V) verglichen. Die Schwellen von 10,5V bzw. 19,5V *1) stellen die untere Toleranzgrenze der Lastversorgung in einer 12V- bzw. einer 24V-Umgebung dar.

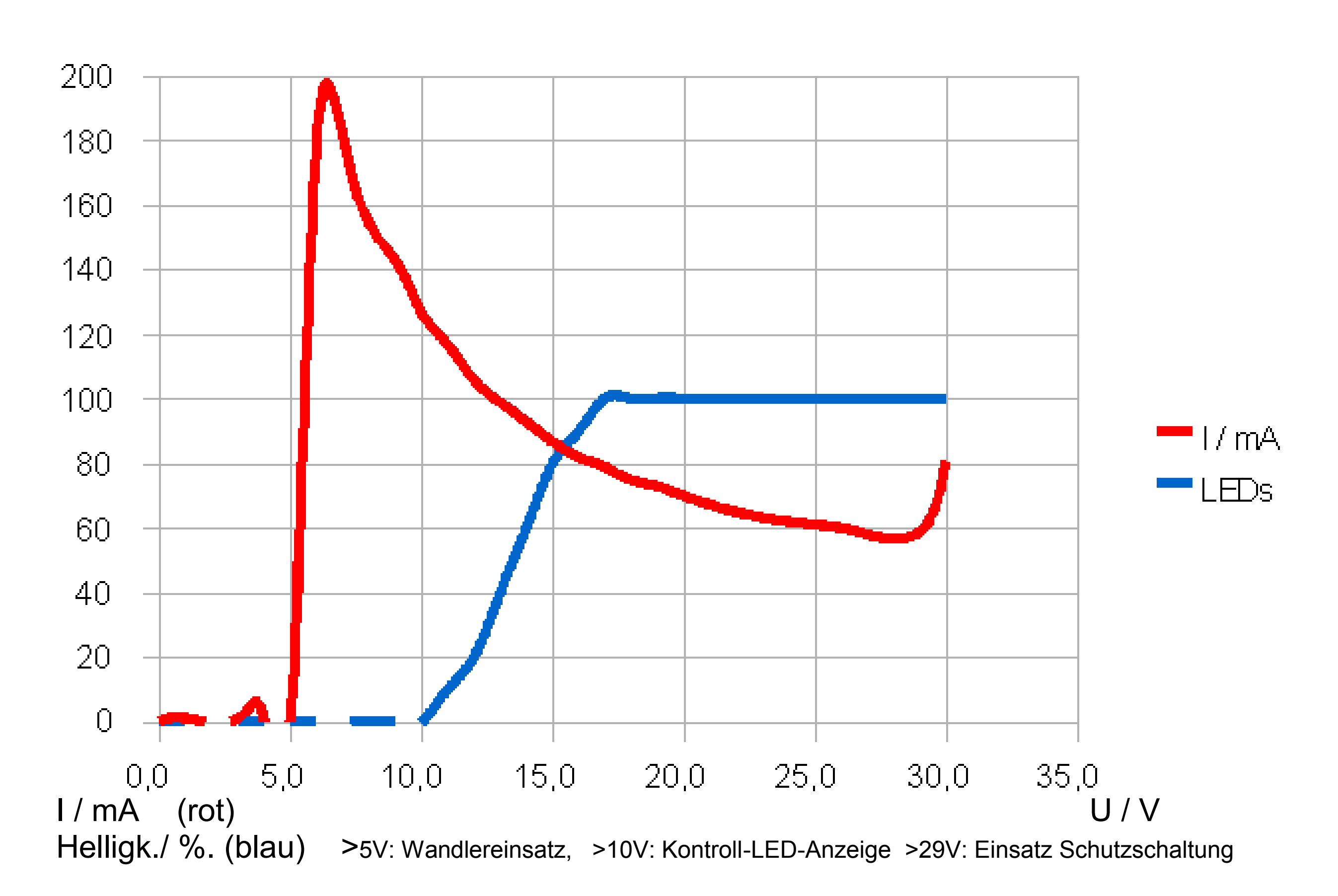


Bild 9: Stromaufnahme und Kontrollanzeige der Einspeisung

LV-Überwachung für 24V-Betrieb *1)

Durch das Einlöten einer Drahtbrücke als Bestückungsvariante wird (mit Teiler dann 37K4 \ 2K32) die Komparatorschwelle von 10,5V auf 19,5V geändert.

Siehe Bild 10 unten und die Abbildung rechts.



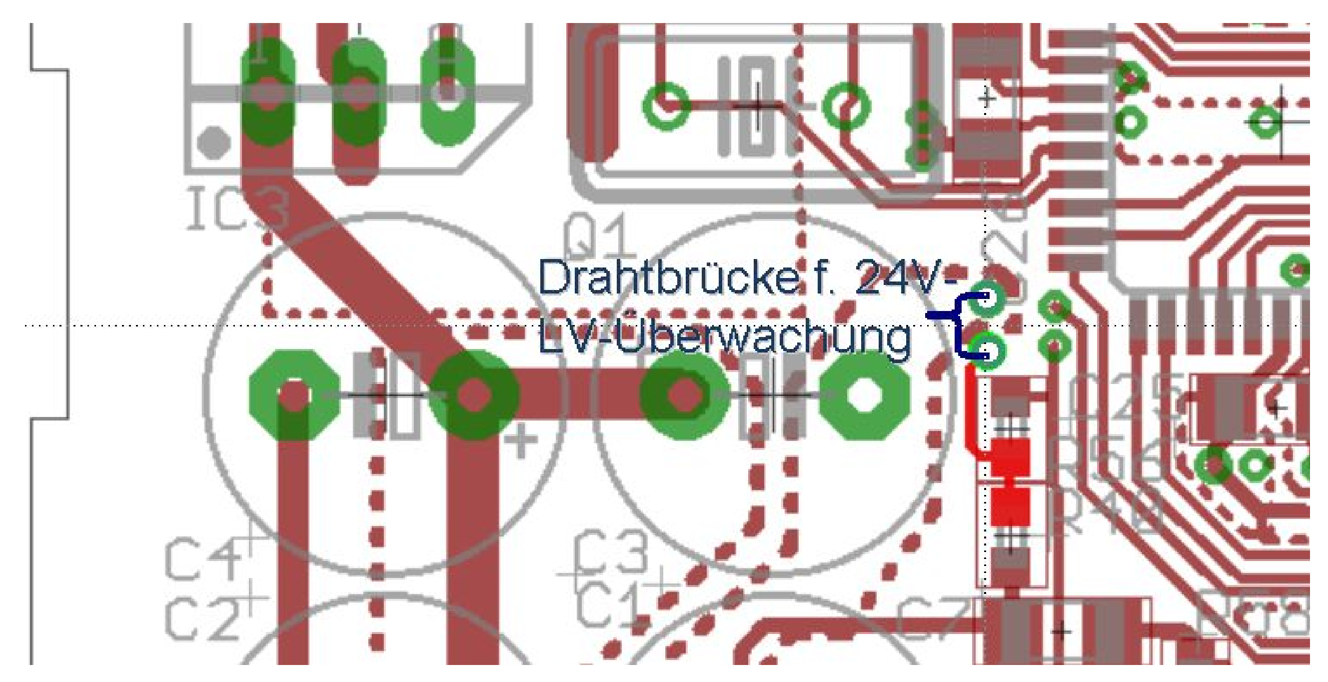


Bild 10: Lage der Brücke für Lastspannungskomparator im 24V-Betrieb (zwischen Prozessor und großem Elko C3; blaue Klammer)

Prozessorversorgung

Sie erfolgt (im gleichen Spannungsbereich 9..28V)

- aus der Lastspannung

oder

- aus der „redundanten Einspeisung“

Hinweis 1: Ist nur die redundante Einspeisung vorhanden (aber nicht die Lastspannung), so fallen lediglich die digitalen Prozessausgänge (DO, Leistungstreiber, H-Schalter) aus. Eingänge, RS-232, Ethernet-Anschluss, small memory card (SMC) etc. bleiben betriebsbereit. Auf diese Weise ist auch "Weiterdenken" trotz Lastspannungsabwurfs (z.B. Notbetrieb, Alarmmeldungen) darstellbar.

Für die interne Prozessorversorgung (mit 5V) wird ein Spannungswandler (9..35V auf 5V) eingesetzt. Diese aufwändige Lösung bietet gegenüber einem einfachen Längsregler — insbesondere bei dem gegebenen weiten (Last-) Versorgungsspannungsbereich — wesentlich geringere Verluste und bessere Pufferzeiten.

Pufferung der Prozessorversorgung nach Ausfall der Einspeisung:

- 150ms bei vorher 24V
- 20 ms bei vorher 12V

Überwachung der Prozessorversorgung

Es gibt keine direkte Überwachung der (5V-) Prozessorversorgung. Sie ist in weiten Bereichen der überwachten Lastspannung gewährleistet. Die sogenannte Brown-out-Funktion des Prozessors kann allerdings einen gezielten Reset anstelle eines „unkontrollierten Sterbens“ der Prozessorfunktion herbeiführen.

3.3 Schutz, Einrichtungen, Anforderungen und Konzepte

Bei Spannungsbereichen, Überlastfestigkeit und -schutz entspricht weAut_01 von der Konzeption her industrieller Automatisierungstechnik. weAut_01 führt keine µController-Pins an Anschlussklemmen bzw. Steckverbindungen — im Gegensatz zu ungeschützten Mikroprozessor- und PC-Baugruppen.

Hinweis / Warnung: Dies gilt nicht für „innen liegende“ Pfostenverbinder und die (ISP-) Programmierschnittstelle, die nur unter Laborbedingungen genutzt oder verändert werden.

Trotz ihrer Robustheit werden das Überschreiten der Grenzwerte und andere Handhabungsfehler die Baugruppe beschädigen.
Hinweise und Werte finden sich in den jeweiligen Kapiteln.

Hier werden jeweiligen Konzepte und Mittel zum Schutz stichwortartig zusammengefasst.

Schutz der Stromversorgungen (LV und red. Einsp.)

Einlötsicherung, Verpolschutzdiode, Transzorp-Diode (Suppessor);

Vorwärtsdioden vor dem Spannungswandler zur Prozessor-Versorgung (mit Vcc, 5V).

Schutz der Prozesseingänge digital und analog

Vorwiderstände, Zenerdiode, Filter-C

Maximale Eingangsspannung:	+/- 60V	(bzw. geltende Kleinspannungsgrenze)
Dauerfestigkeit:	+/- 240V bzw. 240Veff ~	(nicht gestattet, Vorschrift!)

Schutz binärer Prozessausgänge

Treiber-IC + Schutz der Lastspannung (LV, s.o.)

Der Treiber-IC erlaubt (Last-) Ausgangsspannungen bis 30V und Lasten bis (acht mal) 100mA. Die acht Ausgänge des Treiber-ICs ([10]) sind gegen Überlastung und Kurzschluss geschützt; sie sind direkt mit den DO-Ausgangsklemmen verbunden.

Hinweis / Warnung: Das Treiber-IC verträgt keine (Speise-) Spannung über 35V und keinerlei Einspeisung in die (acht) Ausgänge. Desgleichen verträgt es zwar induktive Lasten, aber nicht deren externe, also nicht selbst kontrollierte, Ab- und Zuschaltung.
Ein Verletzen dieser Grenzen und Regeln („eine passive Last pro Ausgang, fest angeschlossen“) kann den Treiber-IC und damit damit die Baugruppe.zerstören.

Weitere Hinweise zu den Grenzwerten des DO-Treibers siehe bei [10].

Schutz der RS232 / V.24 Signale

Der eingesetzte Baustein MAX202CWE bietet 15kV-ESD-protection an den Ein- und Ausgängen (9-pol. SubD). Dies betrifft die vier Signale TX, RX, RTS und CTS am SubD-Stecker; siehe auch das Datenblatt des eingesetzten (MAX-) Bausteins.
Dieser Schutz gilt auch für die auch vorgesehene Verwendung dieser Ein und Ausgänge als „freie“ +/- 6..12V Binärein- und Ausgänge bzw. als Timer/Generator/Counter.

Schutz der Ethernet-Schnittstelle

Galvanische Trennung in der RJ45-Steckverbindung; siehe Bild 11.
PoE (power over Ethernet) ist an dieser Schnittstelle nicht erlaubt.

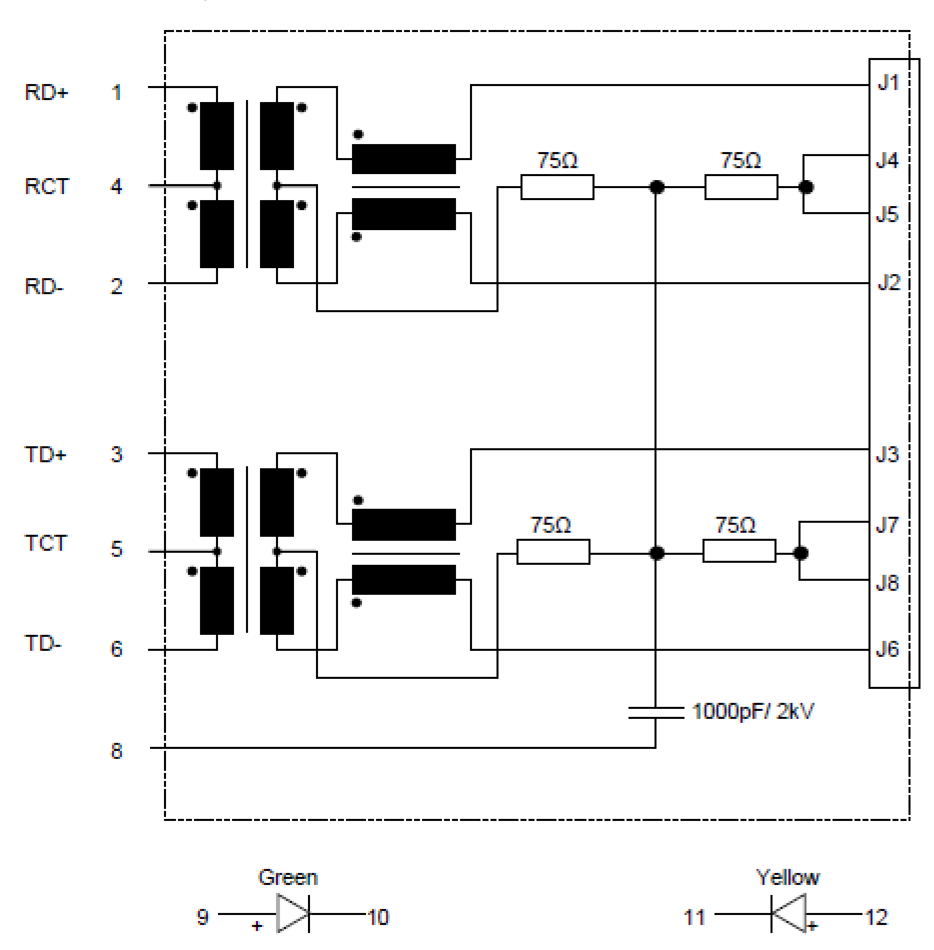


Bild 11: RJ-45-Buchse für den Ethernet-Anschluss mit Trafos, Filtern und LEDs

EMV, Erdung

Die großzügig bemessenen Versorgungsflächen (ground planes) bieten einigen Schutz gegen Ein- und Abstrahlung. Empfohlen wird der Einbau in ein leitfähiges Kunststoff- oder ein mit PE verbundenes Metallgehäuse.

PE (Gehäuse- / Schutzerde) und Gnd (Erde, 0V, Bezugspotential) sind getrennt. Die Spannungsdifferenz soll 30V nicht überschreiten; vor "statischem" auseinander Driften schützt ein Widerstand.
Die Trennung kann mit einem "Jumper" JP2-1=3 aufgehoben werden.

Hinweis: Die Nutzung der seriellen Schnittstelle kann die genannte Trennung von PE und Gnd aufheben, falls diese in der Gegenstelle gebrückt sind. Es gibt auch (billige) "V.24-Kabel", die dies schon ohne Anschluss an eine Gegenstelle normwidrig tun.

3.4 Kommunikationsschnittstelle — Ethernet

Es gibt eine Ethernet-Schnittstelle ([11], [12]) mit

- 10Base-T (10 Mb/s; 100ns bit time)
- Automatic Polarity Detection and Correction
- Full and Half-Duplex
- Zwei Status-LEDs in der RJ45-Buchse

Der Controller ist ein ENC28J60 ([11]). Damit sind auf der Baugruppe praktisch alle transport und application layer Protokolle und Anwendungen implementierbar.
Die durch den Controller 28J60 bedingten Grenzen sind

- 10Base-T (kein 100 / 1G)
- kein Auto cross over

Im Lichte des Mikrocontrollers betrachtet sind dies keine Einschränkungen. Bei Anschluss über einen halbwegs intelligenten switch/router wirken sie sich auch nicht auf das übrige (ggf. schnellere) LAN aus.

Die Ethernet-Schnittstelle ist (naturgemäß) über Transformatoren potentialgetrennt. Die eingesetzte RJ-45-Buchse hat Gleichtaktdrosseln zur Störunterdrückung; siehe Bild 11.

Die (kabelseitigen) Sekundärwicklungen sowie die unbenutzten Stiftpaare sind über Abschlusswiderstände (75 Ohm) und einen Kondensator 1nF/2kV mit PE verbunden. Dieser untereinander verbundene (Gleichtakt-) 75-Ohm.Abschluss aller Adernpaare verbietet nennenswerte Spannungen zwischen den Adern.

Hinweis: Die Baugruppe weAut_01 darf demgemäß nicht an ein Netzsegment mit „power over Ethernet“ (PoE) angeschlossen werden.

MAC-Adresse

Diese ist eine für ein Gerät normalerweise weltweit eindeutige 48-Bit-Adresse, vom Typ IEEE- MAC-48 bzw. nun EUI-48. Die letzten (least significant) 24 Bit sind eine vom Hersteller eindeutig vergebene Schnittstellennummer. Die ersten 24 Bit bezeichnen als OUI die Herstellerfirma. OUIs müssen vom IEEE-RA gekauft werden.

Es gibt mehrere Möglichkeiten damit umzugehen:

1. OUI kaufen und singuläre EUIs vergeben.
   Diese Dienstleistung wird von weinert – automation nicht angeboten.
2. Eindeutiges „locally administered“ Schema verwenden und Gerät zuordnen.

Der zweite Ansatz, der die direkte Interoperabilität der Geräte an einem (Sub-) Netz gewährleistet, ist für (Automatisierungs-) Module in einem lokalen Netz angemessen bzw. hierfür vollkommen ausreichend. Diese beiden Möglichkeiten werden durch das Bit 1 des sechsten (most significant) Byte der MAC-Adresse unterschieden:

0 = globally unique (OUI + serial number)24 Bit
1 = locally administered.

weAutSys erlaubt die Konfiguration der MAC-Adresse und anderer TCP/IP-Einstellungen über die Bedienschnittstelle(n) und deren persistente Hinterlegung im EEPROM (per Bedienung oder via ISP).

3.5 Kommunikationsschnittstelle — V.24 / RS323

Für die Kommunikation via V.24 und optional auch als zusätzliche E/A, insbesondere für Timer / Generator / Counter dient an der Baugruppe eine 9polige SubD-Buchse (Bild 12).

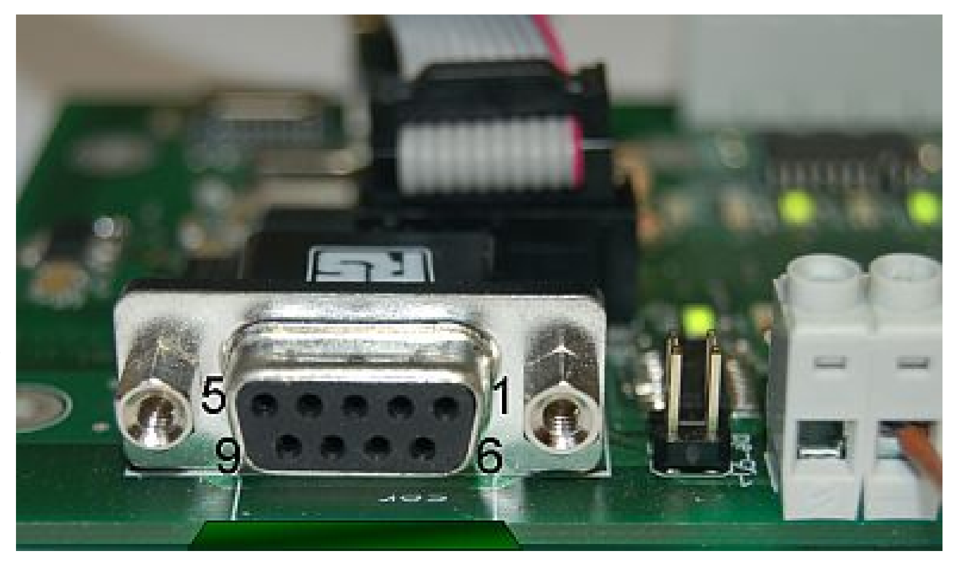


Bild 12:
9 polige SubD-Buchse (female), Blick von außen

Die Buchse (female) ist üblicherweise am DCE (data circuit-terminating equipment, Modem) während der Stecker (male) zum DTE (data terminal equipment, Rechner) gehört. Die Verbindung ist 1:1; die genormten Signalbezeichnungen (Tabelle 13) beziehen sich auf das DTE und damit <u>nicht</u> auf die (DCE-) Sicht der Baugruppe.

Die Pegel sind +/- 3..15 V (log 0 / 1); Pegelwandlung (und Schutz) für vier Signale macht ein entsprechendes IC (MAX202).

Pin	Signal	Sicht DTE	DCE / weAut	Handhabung
1	CD (Hi:Träger da)	Empf	Send	an Lötauge (0,6mm) abgreifbar *1)
2	RxD	Empf	Send	UART-TX via MAX
3	TxD	Send	Empf	UART-RX via MAX; JP1-Pin5 *2)
4	DTR (Hi: DTE ist bereit)	Send	Empf	JP1-Pin1 via R 3K3 *3)
5	Gnd			Signal ground
6	DSR (Hi: DCE ist bereit)	Empf	Send	an Lötauge (0,6mm) abgreifbar *1)
7	RTS (Hi: DTE will send.)	Send	Empf	via MAX an JP1-Pin7 (-PB1/Count.)*4)
8	CTS (Hi: DCE kann empf.)	Empf	Send	via MAX von JP1-Pin 9 (-Timer/PD6-) *5)
9	RI (Hi: Call)	Empf	Send	an Lötauge (0,6mm) abgreifbar *1)

Tabelle 13: SubD-Steckverbindung für RS232 (V.24) – Belegung

Anm. *1): Freie Verwendung, i.A. aber unbelegt / unbenutzt.

Anm. *2): Für Ponyserver-ISP "reset=!txd" (siehe unten, Bilder 14 und 15, Seite 23)

Anm. *3): Für Ponyserver-ISP "mosi=dtr" (siehe Bilder 14 und 15, Seite 23)

Anm. *4): Brücken von JP1-Pin7=Pin8 (Bild 14) legt RTS (7) an Port B1 (Eingang);
Verwendung ist
a) Abfrage des RTS für V.24-Kommunikation (Flusskontrolle) oder
b) SubD Pin 7 als Zählereingang (Timer/Counter 1 External Counter Input)

Anm. *5): Brücken von JP1-Pin9=Pin10 (Bild 14) legt CTS (8) an Port D6 (Ausgang);
Verwendung ist
a) CTS-Flusskontrolle des V.24-Empfangs
(u.U. sinnvoll wegen begrenzter Puffergröße)
b) SubD Pin 8 als Generatorausgang
(PD6 - Timer/Counter2 Output Compare Match B Output)

Anm. *3): Zahlreiche Verwendungsmöglichkeiten; siehe unten.

Pegelanpassung (V.24 <-> TTL) und Schutz (ESD) macht für die vier Signale an den Pins 2, 3, 7 und 8 der SubD-Buchse ein Maxim-Baustein (MAX 202).

Das Laufzeitsystem weAutSys verwendet die V.24-Schnittstelle in Grundeinstellung als Standardein- und -ausgabe für Befehle und Log-Ausgaben mit:
38400 baud, 8 Databits, 1 Stopbit, no parity, no flow control.

Über die (standardmäßig gesteckten) Brücken JP1-7=8 und JP1-9=10 stehen RTS und CTS an den Ports PB1 bzw. PD6 zur Verfügung. Optional kann also CTS und/oder RTS flow control genutzt werden (in weAutSys jeweils unterstützt).

Alternativ können statt CTS oder RTS auch beliebige Port-Funktionen (siehe auch folgendes) genutzt werden. Das gilt grundsätzlich auch für RxD und TxD.

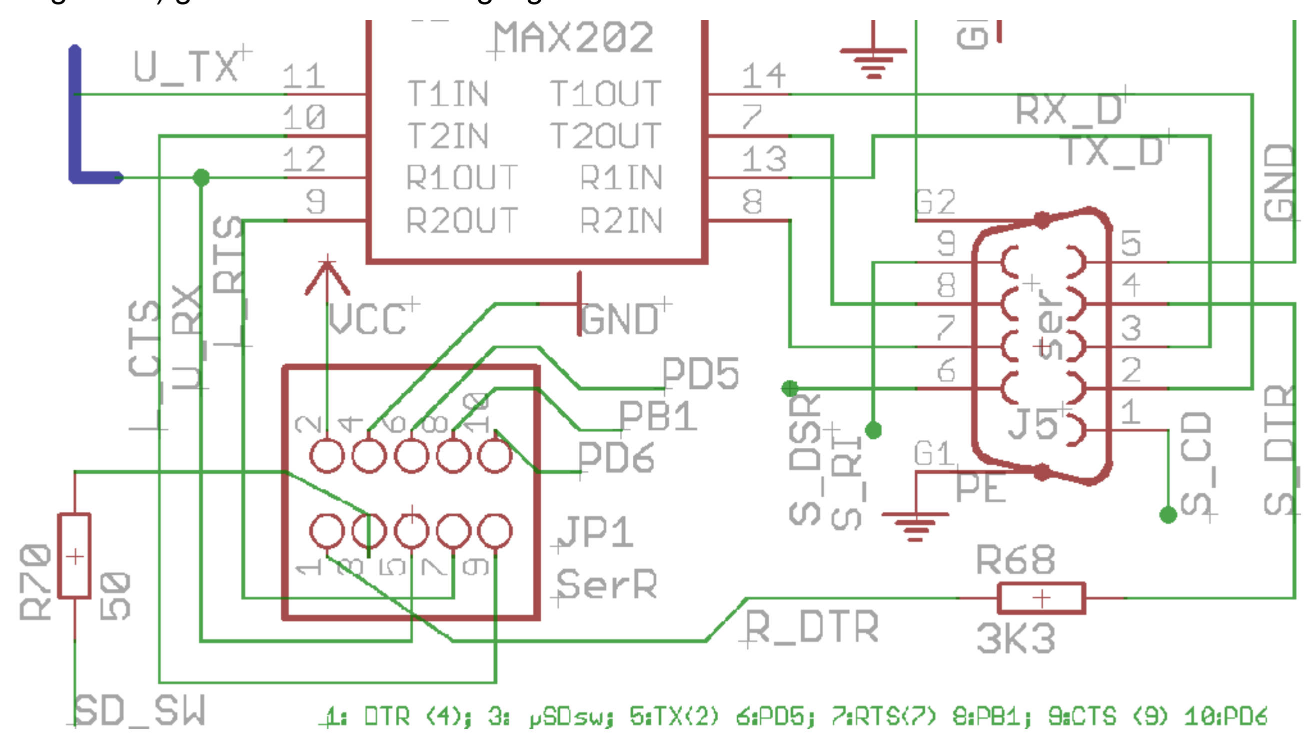


Bild 14: Anschluss V.24, timer counter generator (PD6, PB1) und Rangierung JP1

Die Belegung der Rangierstiftleiste JP1 passt zur ISP/SPI-Schnittstelle; vgl. die Bilder 14 und 15 auf Seite 23 und siehe unten.

Counter / timer / generator

Die an die Siftleiste JP1 geführten Ports PD5 (JP1-Pin6), PB1 (JP1-Pin8) und PD6 (JP1-Pin10) können von da *) auch als freie Ports für interne Erweiterungen genutzt werden.

Hinweis *): Bei einer solchen Verwendung ans freie Ports sind diese (elektrisch) ungeschützt. Insofern dürfen diese dann lediglich für interne Erweiterungen verwendet werden. Sie sind nicht als Prozess-I/O herauszuführen.

Bei Verwendung von PB1 als Eingang und PD6 als Ausgang können diese über den MAX202-Baustein invertiert mit V.24-Pegeln ESD-geschützt auf die SubD-Buchse geführt werden. Dies geschieht mit zwei Steckbrücken auf der Stiftleiste JP1; siehe dazu Tabelle 13 mit Erläuterungen sowie Bild 14).

Eine vorgesehene Verwendung dieser Ports ist

- PB1 als Zähler. (8 bit) = Timer1 **) mit optional Clock - Eingang
- PD6 als Generator / Zeitgeber (16 bit) = Timer2 mit optional compare/match B-Ausgang
- PD5 (evtl.) Schalter des SMC-Schachts (= JP1-3; Lo : small memory card inserted)

Hinweis **): Timer 0 (8 bit) dient bei weAutSys intern fest vergeben als Systemzeitgeber (tick interrupt). Die Timer 1 und 2 stehen der Anwendungssoftware zur freien Verfügung

3.6 Programmierung und Erweiterung

SPI

Der ATmega1284 (644 etc.) bietet zwei SPI-Schnittstellen:

- SP1 mit PB5..7=MOSI,MISO,SCK (standard SPI /ISP) und
- SPI2 mit PD2..4=MISO,SKC,MOSI (UART as SPI / master only)

Beide werden genutzt:
SPI2:

- für den Ethernet controller
- für die SMC (Steckplatz / slot für kleine Speicherkarten)

SPI1:

- als ISP
- für die digitalen Ausgänge (DO)
- für die LEDs der (digitalen) Eingänge (DI-LEDs) sowie
- für mögliche Erweiterungen.

SPI1 wird für die vorhandene Peripherie (DO, DI-LEDs) mit maximaler Geschwindigkeit (10 MHz = ½ Prozessortakt) betrieben. Bei SPI1 werden MISO, MOSI und SCK widerstandsentkoppelt für eingebaute Peripherie verwendet, um mit der ISP-Funktion nicht zu kollidieren.

SPI als ISP

Für die Verwendung als SPI-Erweiterung vor allem aber als ISP-Schnittstelle wird SPI1 in der übliche 10-Pin-Variante (statt 6-Pin) auf einen Wannenstecker geführt, siehe Bild 15 auf Seite 23. I.A. wird der 10-polige Wannenstecker „ISP“ nur bei Softwareupdates genutzt und dann über ein Programmiergerät (wie z.B. smartUSB) oder über die serielle Schnittstelle (via Pfostenstecker JP1; s.u.) mit dem Entwicklungsrechner verbunden.

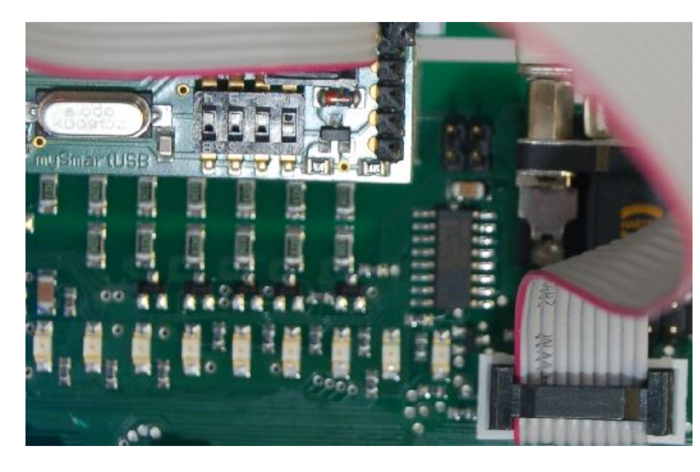
2	VCC	Gnd*)	Gnd	Gnd	Gnd	10
1	MOSI	Gnd	/Reset	SCk	MISO	9

Bild 15: Belegung des ISP-Schnittstellen-Steckers (10pol. Wannenstecker)

Hinweis: Pin 3 (Gnd) ist so üblich, obgleich von Atmel als NC (unverbunden) definiert.

Hinweis: Der ISP-Wannensteckers kann als SPI für Erweiterungsmodule genutzt werden. Dies funktioniert zusammen mit entsprechender alternativer Verwendung von 2wire (s.u.) oder der Ports PB1, PD5 und PD6 (s.o.).

Warnung: Die vorgegebene (Standard-) Belegung der 10 Stifte für 6 Signale (Bild 15) ist schlecht gewählt. Ein Voreilen von Gnd ist kaum machbar, ein Nacheilen sogar wahrscheinlich. Ziehen und Stecken unter Spannung ist also eher kritisch zu sehen.
Die hier nicht verwendete 6-polige (Bild 17) Belegung wäre diesbezüglich noch kritischer.



Via Programmiergerät,
hier mySmartUSB als Beispiel,
zum Entwicklungsrechner

Bild 16: ISP an Programmiergerät

Für die Verwendung des Programmiergeräts mySmartUSB muss für Windows der Treiber von Silicon Labs (CP210x USB to UART) installiert sein. Mit dem Befehl

```
avrdude -p atmega1284p -P com7 -c avr911  -n -v
```

als Test werden die Einstellungen des µControllers ausgelesen aber nichts programmiert. Statt com7 im Beispiel ist der jeweils verwendete COM-Port einzusetzen.

2	VCC	MOSI	Gnd	6
1	MISO	SCK	/Reset	5

Bild 17: Alternative 6polige Belegung des ISP-Schnittstellen-Steckers; so z.B. beim Programmierer mySmartUSB light.

Hinweis: weAut_01 verwendet diese Belegung nicht.

Bei Nutzung des kompakteren (und schnelleren) Programmiergeräts mySmartUSB light benötigt man einen Adapter von dessen 6poliger ISP-Belegung (Bild 17) zur hier verwendeten 10poligen (Bild 15).

Der entsprechende Testbefehl mit mySmartUSB light lautet dann sinngemäß

avrdude -p atmega1284p -c avrisp2 -P com4 -n -v

ISP (SPI) via direktem seriellem Anschluss

Alternativ kann die ISP-Schnittstelle auch ohne zwischengeschaltetes Programmiergerät mit der seriellen Schnittstelle des Entwicklungsrechners verbunden werden.
Solche Verfahren laufen unter Begriffen wie "[serial] bit banging" oder "Pony".

```
# serial ponyprog (weAut_01 direkt)
programmer
  id    = "ponyWeAut";
  desc  = "weAut serial, reset=!txd sck=!rts mosi=dtr miso=!cts";
  type  = serbb;
  reset = ~3;
  sck   = ~7;
  mosi  = 4;
  miso  = ~8;
```

Listing 18: Konfiguration für "Pony direkt"
(in C:\Programme\WinAVR\bin\avrdude.conf bzw. /etc/avrdude.conf)

Diese Programmiermöglichkeit kann in der Konfiguration gemäß Listing 18 ohne zusätzliche Hardware genutzt werden, wenn jeweils die Pins 1, 5, 7 und 9 vom Wannenstecker "ISP" und der Rangierstiftleiste JP1 verbunden werden; vgl. die Bilder 14, 15 und 20. Das 1 zu 1-Verbinden aller 10 Pins (mit dem üblichen Programmierflachkabel) ist einfachkeitshalber ebenso möglich; siehe Bild 19.

9-pol. SubD: V.24 zum Entwicklungsrechner

10-pol. Flachband 1 zu 1 – Verbindung
ISP = JP1

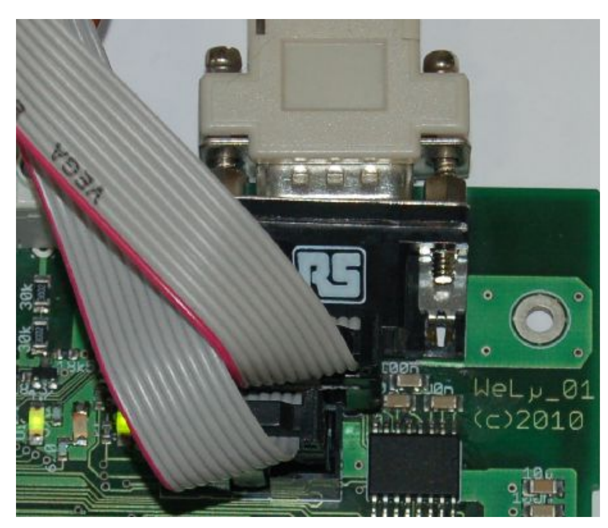


Bild 19: ISP an V.24

Pony programming
ohne Zusatzhardware

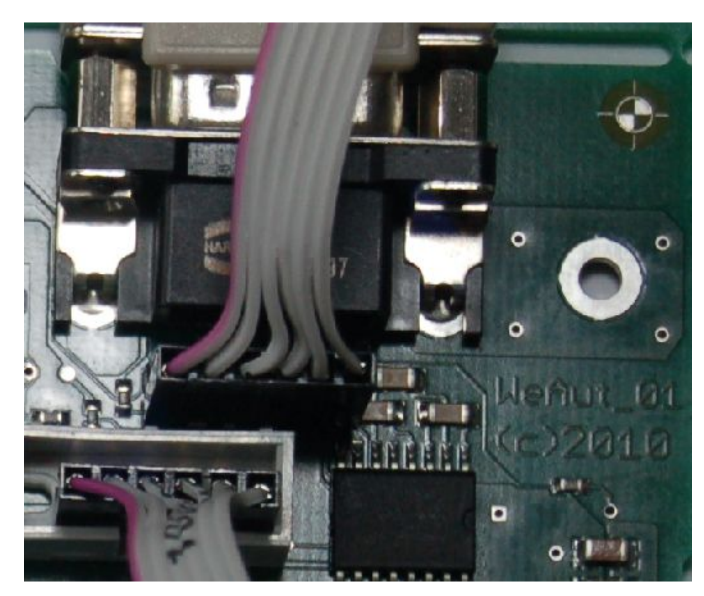
Der Vorzug der direkten Verbindung ist in der Einsparung eines zusätzlichen Programmiergeräts. Der häufig behauptete Geschwindigkeitsnachteil war hingegen, im Vergleich mit dem smartUSB, entweder

- bei manchen Systemen und USB-zu-seriell-Adaptern nicht zu beobachten — eher das Gegenteil (Windows Server 2003, Linux Mint 12 mit /dev/ttyS0)

oder aber

- so gravierend, dass alles außer fuses Lesen und Setzen praktisch unmöglich war (Windows 7 mit USB-Adapter).

Hinweis: smartUSB light ist nicht nur schneller als smartUSB, es kann auch fuses programmieren. Er hat aber die 6polige Schnittstelle und wird leider ohne Adapter auf 10polig geliefert



Seriell zum Entwicklungsrechner

"Bitbang" oder "Pony" programming ohne zusätzliches Programmiergerät

Einreihiges. Flachband 1 zu 1 - Verbindung ISP = JP1, siehe auch Bild 21

Es genügt, die unteren 5 Pins von ISP und Pfostenverbinder JP1 zu verbinden

Bild 20: ISP an V.24 (einreihig)

Rangierung / Erweiterung (Stiftleiste JP 1)

2	VCC	Gnd*)	PD5	PB1	PD6	10
1	DTR	/SMCin	/RX	/RTS	/CTS	9

Bild 21: Belegung der Stiftleiste JP1 (oberhalb 10pol. Wannenstecker)

Hinweis: Untere Reihe 1 zu 1 mit unterer Reihe des ISP-Wannenstecker für serielle ISP verbinden; vgl Bilder 14 und 20.

"DTR" (Pin1) geht über Schutzwiderstand (3K3) direkt an die SubD- (V.24) Buchse und ist nur für "ISP via serielle Schnittstelle" gedacht. "/RTX", "/RTS" und "/CTS" sind hingegen (via MAX Baustein) invertiert und pegelgewandelt.

ISP über diese serielle Schnittstelle mit dem bootloader von weAutSys

Am wenigstenaufwändig ist die Änderung der System oder Anwendungssoftware mit dem in weAutSys integriertem bootloader. Nur dessen Laden/Ändern selbst erfordert einen der vorher beschriebenen Wege.

Two-wire-interface / I²C

Das (komplexe) 2wire- oder I2C-Protokoll wird von über 1000 ICs und vielen Zusatzmodulen, wie LCD-Displays, verstanden. Der I2C ist in üblicher Belegung an der Stiftleiste JP3 (Bild 22) verfügbar. Da dieser Anschluss die (5V-) Versorgung mitführt, kann er auch gleich eine Erweiterungsbaugruppe speisen.

SDA______________JP3_Pin1____PC1

VCC______________JP3_Pin2

Gnd______________JP3_Pin3

SCL______________JP3_Pin4____PC0

Bild 22: Belegung der 2wire-Stiftleiste ([6])

Alternative Verwendung

Mit JP3 lassen sich auch PC0 und PC1 als Ports auf eine Erweiterungsbaugruppe zur dortigen beliebigen Verwendung führen. Beispielsweise lassen sich DCF77-Empängermodule mit einen Steuereingang (meist enable) und einem Datenausgang hier anschließen.

Wird auch die ISP-Schnittstelle als SPI (Bild 22) auf diese Erweiterungsbaugruppe geführt, können dort auch zwei weitere SPI-Module (mit PC0 und PC1 als CS) betrieben werden.

Wird diese Schnittstelle weder als SPI noch für solche Erweiterungen verwendet, können JP3_Pin1 (PC1) als Test-Pin_1 und JP3_Pin4 (PC0) als Test-Pin_2 für die Entwicklung bzw. Beobachtung der Software konfiguriert und Verwendet werden. Das ist eine Art "messtechnisches Äquivalent" zu den Test-LEDs (rt und gn) und wird in weAutSys für die Softwareinstrumentierung so unterstützt.

Kein JTAG

Die Verwendung der JTAG-Schnittstelle (des Prozessors) ist nicht vorgesehen. Hierfür würden die Ports PC2 bis PC5 benötigt; sie sind anderweitig fest belegt; vergleiche Tabelle 26 ab Seite 32. Das Mengengerüst erlaubt es auch nicht, auf vier Ports zu verzichten.

Hinweis: Für das Funktionieren der Baugruppe ist es unabdingbar, die "JTAG-fuse" entsprechend zu setzen (Fuse High, Bit 6, JTAGEN auf 1 = disable).

Small memory card (SMC)

weAut_01 hat einen Steckplatz für SMC (kleine Speicherkarten) am unteren rechten Rand, welche ausschließlich die SPI-Schnittstelle nutzt (SPI2 und PC6 als \CS).

Write protect- Schalter (Schieber) und andere Mechanismen sind bei passenden SMCs nicht vorgesehen.

Allerdings hat der eingesetzte SMC-connector (DM3AT) einen Schalter, der bei eingesetzter Karte JP1-Pin 3 (über 50 Ohm) an Gnd legt, also ein "card detection switch. Dieser kann bei Bedarf von dort auf einen freien Port, vorzugsweise PD5, gebrückt werden; vgl. dazu das Bild 14 auf Seite 21 sowie Bild 21 auf Seite 25.

Das Abfragen dieses Ports kann den Aufwand des probehalber Ansteuerns einer Speicherkarte nur zur Feststellung, dass gar keine gesteckt ist, ersparen. Bei Verwendung von SMCs sollte die Brücke auf PD5 (JP2 3=6) gesetzt werden – ggf. auch als Draht oder SMD-Widerstand auf der Baugruppenunterseite eingelötet.

SMCs sind gerade wegen ihrer SPI-Schnittstelle bestens für µController mit SPI als Speichererweiterung geeignet.

Pin	Name	Signal Function
1	NC	No Connect
2	/CS	Chip Select
3	DI	Master Out/Slave In (MOSI)
4	Vdd	Supply Voltage 2.7V .. 3.6V
5	CLK	Clock
6	Vss	Ground
7	DO	Master In/Slave Out (MISO)
8	RSV	Reserved

Tabelle 23:
SMC - Signale im SPI-Mode

[52] ist eine gute Beschreibung der Grundlagen zur Handhabung einer SMC, es beschreibt (als Standard) dies und die Basis-Kommunikation, Kartenregister und dergleichen ausführlicher.

3.7 Prozessschnittstellen
Binäre Prozesseingänge / digital input DI

Anzahl: 8
Spannung: 24 V / 12V kompatibel mit einzeln — d.h. kanalweise — konfigurierbaren Schwellwerten bzw. Hysteresen:

H)	1	über 10,9 V	0	unter 9,0V	(narrow, **H**igh)
L)	1	über 6,1V	0	unter 3,3V	(narrow, **L**ow)
W)	1	über 10,9 V	0	unter3,3V	(**W**ide hysteresis)

weAutSys unterstützt diese drei möglichen Betriebsarten und zusätzliche Filteroptionen

Eingangsimpedanz: 60 kOhm

Filter / Tiefpass: 66,5µs

Erlaubter Eingangsspannungsbereich: +/- 60V (Kleinspannung)

abs. max. rating : +/- 250V bzw. 250Veff ~ (zerstörungsfrei)

Hinweis, **Warnung**: Schaltung und Layout verteilen den Eingangsspannungsabfall auf zwei Widerstände und verwenden relativ viel Platz für weite Abstände untereinander und zu empfindlichen Bauelementen.
Dennoch **verbieten** geltende Vorschriften zu Luft- und Kriechstrecken i.A. mehr als +/-60 V an den Eingangsklemmen. Halten Sie diese Sicherheitsvorschriften streng ein.

Analoge Prozesseingänge / analogue input AI

Jeder der 8 (digitalen) Eingänge kann auch als Analogeingang genutzt werden. Hier sind die Möglichkeiten und die Genauigkeit gegenüber den grundsätzlichen Möglichkeiten des µControllers jedoch etwas eingeschränkt, da die Schaltung dafür nicht primär ausgelegt ist. Die Einschränkung bestehen in der 2%-Genauigkeit der Eingangsteiler, der (auch nutzbringend einsetzbaren) Nichtlinearitäten der Schutzdioden und dem Fehlen einer getrennten "analogue ground plane".

Für sehr viele Anwendungen ist eine solche (einfache) AI jedoch vollkommen hinreichend.

Eingangsteiler: * 0,27 (+/-2%)

Filterzeitkonstante: 66,5µs (f_g = 15kHz)

Messbereich	**A**:	0..+4V	10 Bit Auflösung, „single ended"-Verwendung eines Kanals interne 1,1V-Referenz
	B	0..+10V	dto., dto., mit der internen 2,5V-Referenz (Bild 24, Seite 29)
	C	0..+18V	dto., dto., mit VCC = 5V als Referenz; hier wird das obere Ende des Bereichs von der Schutzdiode beschnitten (Bild 25).

Andere Messbereiche sind mit zusätzlicher äußerer Beschaltung oder durch die Verwendung von zwei Kanälen als Differenzeingang und dem internem Verstärker (des µControllers) denkbar, aber i.A. weniger sinnvoll.

Die Bilder 24 und 25 zeigen für die o.a. Bereiche B und C der Zusammenhang der analogen Messung bzw. AI-Antwort mit der Eingangsspannung. Bei sinnvoller Beschränkung auf die oberen 8-Bit (der 10Bit-Auflösung) zeigt sich eine gute Übereinstimmung der Kanäle ebenso wie auch eine stabile Rate Volt pro digit (untere grüne Kurve), solange die Eingangsspannung unterhalb des Einsatzes der (DI-) Schutzschaltung / Begrenzung bleibt.

Der für reine Messzwecke an sich störende Einsatz dieser Begrenzung (oberhalb von etwa 10V, Bild 25) kann auch nützlich sein, wenn niedrige Werte mit guter Auflösung und Genauigkeit erfasst werden sollen und man zusätzlich einen großen „Overflow-“ Bereich hoher Werte unterscheidbar erfassen möchte.

Der Bereich C (Bild 25) ist bei Lastspannung 12V bzw. 24V und mit einer entsprechenden Beschaltung von Namur-Gebern für Kurzschluss- und Drahtbrucherkennung geeignet.

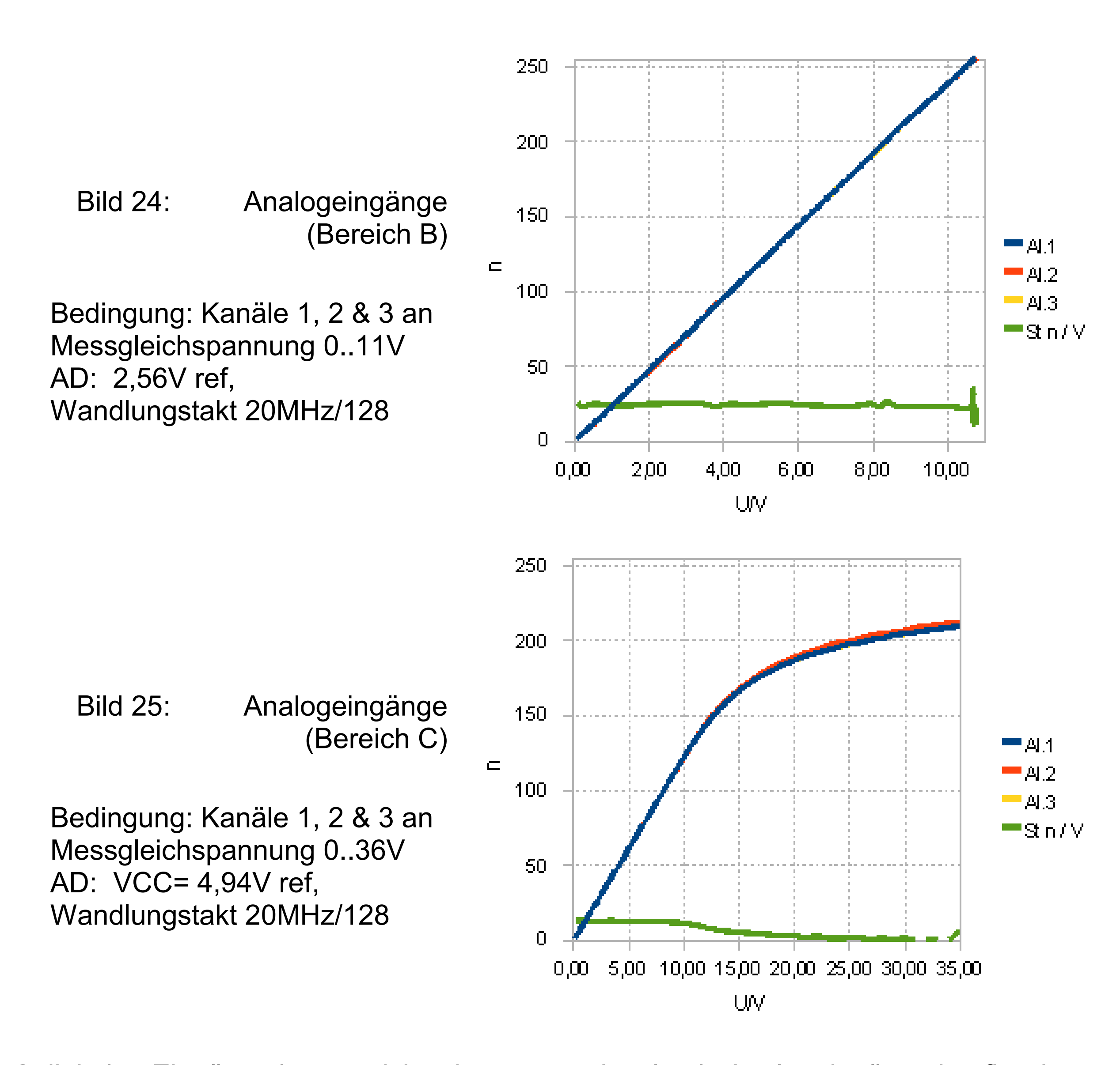


Bild 24: Analogeingänge (Bereich B)

Bedingung: Kanäle 1, 2 & 3 an
Messgleichspannung 0..11V
AD: 2,56V ref,
Wandlungstakt 20MHz/128

Bild 25: Analogeingänge (Bereich C)

Bedingung: Kanäle 1, 2 & 3 an
Messgleichspannung 0..36V
AD: VCC= 4,94V ref,
Wandlungstakt 20MHz/128

Alle 8 digitalen Eingänge lassen sich, wie gesagt, einzeln als Analogeingänge konfigurieren; dies wird von weAutSys unterstützt. Bei gleichem Bereich und der bei Bild 24 und 25 angegebenen Wandlungsrate sind alle 8 Kanäle in weniger als einer Millisekunde abgearbeitet. Die Anwendungssoftware kann bei Bedarf also jeweils aktuelle Analogeingangswerte in ihrem 1ms-Zyklus verarbeiten.

Binäre Prozessausgänge / digital output DO

Anzahl:	8
Logisch 1 " = An" :	Lastspannung - 0,8V
Logisch 0 = "Aus" :	Hi-Impedanz (evtl. geringfügiger Reststrom)
Laststrom:	100 mA max. pro Ausgang
Laststomsumme:	800 mA (max) für alle 8 Kanäle gemeinsam
Lastarten:	Geeignet für ohmsche und induktive Last.
Schutz:	Absteuerung bei Überlastung (Kurzschluss) und Übertemperatur durch das Treiber-ICs ([10]).
Fehlererkennung:	Der Überlastfall wird als Fehlerzustand angezeigt (LED 8 und Software). Aus dem Fehlerzustand kommt man durch disable / enable des Treiber-ICs wieder raus.

In den Grenzen der kurzzeitigen Überlastbarkeit (d.h. ohne in den Fehlerzustand mit Abschaltung zu geraten) sind auch Glühlämpchen (Signallampen z.B.) ansteuerbar. Deren etwa um den Faktor 10 erhöhter Einschaltstrom wird i.A. einen Vorwiderstand erfordern, um die sofortige Überstromabschaltung zu vermeiden.

Counter / timer / generator

Dies stellt eine der alternativen Verwendungen der SubD-Buchse und des Schutz- und Pegelwandlungsbausteins (MAX202), also der V.24 / RS232-Schnittstelle bzw. von Teilen von ihr, dar. Näheres siehe dort auf Seite 22.

Mit Verzicht auf die serielle Kommunikation stehen hier auch zwei Ein- und zwei Ausgänge mit vielfältigen Verwendungsmöglichkeiten zur Verfügung.

4. Softwareentwicklung

Diese Kapitel behandelt die grundsätzlichen Möglichkeiten und Randbedingungen für die Software der Baugruppe unabhängig von der Wahl eines bestimmten Laufzeit- / Betriebssystem wie weAutSys.

weAutSys ist natürlich genau für diese Bedingungen gemacht bzw. hierfür konfigurierbar (`MCU = atmega1284p , PLATFORM = weAut_01`). Im Allgemeinen ist es empfehlenswert, die Anwendungssoftware auf der Basis einer weAutSys-Lizenz zu entwickeln.

4.1 Hardware-Randbedingungen Prozessor-Grundeinstellungen (Fuses)

Variante 1, robust:	l fuse F7	hfuse D9	efuse FD
Variante 2, stromspar.:	lfuse FF	hfuse D9	efuse FD

Ein stabiler Lauf des (Quarz-) Oszillators ist absolut "spielentscheidend".
Variante 1 unterscheidet sich hier lediglich durch die "robustere" Einstellung (full swing).

Port-Verwendung

Die AVR-Controller ATmega164A, ATmega164PA, 324A, 324PA, ATmega644A, 644PA und der bei weAut_01 eingesetzte ATmega1284 haben vier weitgehend frei konfigurierbare 8-Bit-Ports **PA** bis **PD**. Die Freiheit ist insofern eingeschränkt, als die verwendeten integrierte I/O-Subsysteme, wie UARTs, SPIs, timer/counter, AD, Komparator, two-wire-interface und so weiter, an bestimmte Port-Pins gebunden sind.

Alle 32 Port-Anschlüsse sind als digitale Aus- oder Eingänge mit zuschaltbarem Pull-Up-Widerstand nutzbar, sofern keine ihrer jeweiligen eben erwähnten Sonderfunktion verwendet wird. Die Schaltung der Baugruppe weAut_01 nutzt viele dieser Sonderfunktionen fest, schließt einige aus und sie erlaubt bei manchen die optionale Nutzung der jeweiligen Sonderfunktionen oder als freie Ports für Erweiterungen.

Fest genutzte Sonderfunktionen

SPI (PB5..7):	ISP, Master-SPI für Baugruppenperipherie (DO, DI-LEDs)
SPI2 (UART2, PD2..4):	Master-SPI für Ethernet-controller und SMC
UART (PD0..1):	serielle Schnittstelle (V,24 / RS232)
Komparator (PD3):	für die Überwachung der Lastspannung

Optional nutzbare Sonderfunktionen

Two wire (PC0..1):	2wire / I²C-Erweiterung	
Timer 1 (PB1)	Zähler	auch invertiert via RTS-Eingang, SubD
Timer 2 (PD6)	Generator	auch invertiert via CTS-Ausgang, SubD

Ausgeschlossene Sonderfunktionen

JTAG (PC2..5):	Ports sind für andere Zwecke fest belegt.

Pin	Verwendung	Alternative	Anmerkung
Port A	Digitale Eingänge 0..70V	Analog-Ein	+/- 240 V abs. max, rating
PA0	DI 0	AI 0	Eingangsteiler Filter Schutzdiode
PA1..6	DI 1..6	AI 1..6	
PA7	DI 7	AI 7	
Port B			
PB0	Taste „Enter“ (0: gedrückt)		frei verwendbar
PB1	Eingang RTS, Eingang Zähler 1	freier Port	JP1 Pin 8; rangierbar
PB2	LED gn (LED 19)		frei verwendbar
PB3	Komparatoreingang Lastspannung		Eingangsteiler
PB4	LED rt (LED 18)	frei verwendbar	Lastspannung weg (Komp.)
PB5	SPI1 MOSI		ISP direkt, aber andere SPI-Geräte entkoppelt (jew. 270 Ohm)
PB6	SPI1 MISO		
PB7	SPI1 SCK		
Port C			
PC0	TwoWire (I²C) SCL	freier Port	JP3 Pin 4 testpin1
PC1	TwoWire (I²C) SDA	freier Port	JP3 Pin 1 testpin0
PC2	/CS für DI-LEDS (Register)		mit SPI1, Übernahme mit positiver Flanke
PC3	/CS für DO (Register)		
PC4	Eingang DO /fault		und LED rt
PC5	DO enable (1) und LED (rt, 0))		LED "DO disable"
PC6	MicroSD /CS		Pegelanpassung 3V; SPI2
PC7	Ethernet /CS		
Port D			
PD0	UART0 RX		UART via MAX202
PD1	UART0 TX		
PD2	SPI2 (UART1) MISO		SPI für Ethernet und small memory card (SMC)
PD3	SPI2 (UART1) MOSI		
PD4	SPI2 (UART1) SCK		
PD5	MicroSD /inserted (JP1-3=6)	freier Port	JP1 Pin 6
PD6	Ausg. CTS, Ausg. Timer/Generator 2	LED gn, Port	JP1 Pin 10, rangierbar
PD7	Ethernet INT (Int31)		

Tabelle 26: Port-Verwendung

4.2 Grundsätzliche Möglichkeiten

Es gibt Werkzeuge für die Programmierung des µControllers in Assembler, C, ja sogar in C++, in der exotischen Sprache "BasCom" und sicher für manche andere Programmiersprache mehr.

Sprache

Der hier verfolgte, empfohlene Ansatz ist die durchgängige (ausschließliche) Verwendung von **C**. Dies und die Vermeidung von Assembler ist gängige Industriepraxis im embedded Bereich. Für die AVR Atmel controller ist die GNU Werkzeugkette (GNU, GCC) als AVR-GCC bzw. WinAVR geeignet.

Der GNU-C-Compiler für AVR und die zugehörigen Bibliotheken machen ihre Sache weitgehend richtig gut — mit ganz wenigen (teilweise bösen) "Ausrutschern". Insofern kommt man um das Lesen (.s-files) des Assembler-Ergebnisses nicht herum. Das meiste lässt sich dann mit Umformulieren der C-Quelle schrittweise verbessern. Wenn denn doch punktuell signifikante Verbesserungen durch Assembler möglich und nötig sind, ist inline assembler in der C-Quelle das Mittel der Wahl und ohne Bruch der Werkzeugkette möglich.

Plattform, Werkzeuge

Diese GNU Compiler lassen sich auch mit zeitgemäßen leistungsfähiger und auch im professionellen Bereich genutzter Entwicklungswerkzeugen, wie SVN, Doxygen und Eclipse nutzen bzw. und in diese integrieren.

Für die serielle Kommunikation mit dem Modul hat sich das Programm HTerm bewährt.

Für die Analyse des Netzwerkverkehrs (Ethernet) ist Wireshark (am besten ergänzt mit einem sog. break out switch) geeignet und bewährt.

Alle diese Softwarewerkzeuge sind frei von Lizenzgebühren als download erhältlich. Erfolgreiche Installationen und Nutzungen gab unter (jeweils professionell) **Windows** Server 2003, Windows XP sowie Windows 7 (32) und Windows 7 (64), wobei letztere die meisten Probleme machen.

Relativ problemlos ließ sich die ganze Kette auch unter **Linux** Mint (12, 14) aufbauen und nutzen. Hier versagten lediglich die Doxygen binaries; Doxygen musste hierfür aus den Quellen neu gebaut werden.

Der Vorzug der Windowsplattform ist die Nutzung des in vielen Entwicklungsabteilungen genutzten und dort für viele andere Werkzeug notwendigen Quasi-Standards. Der Vorzug von Linux ist die nun vollkommene Freiheit von Lizenzgebühren – und dass dort Wireshark vernünftig läuft.

Es ist auch möglich, größere versionsverwaltete (SVN) Projekte abwechselnd bzw. gleichzeitig unter Windows- und der Linux-Plattformen zu bearbeiten. Der Preis hierfür ist etwas zusätzlicher Sorgfalt bei Konfigurationsdateien sowie das doppelte Bereitstellen von ein / zwei Skripten, also xyz.bat und xyz.sh, die dann (versionsverwaltet) konsistent zu halten sind.

Hinweis: Konkrete Hinweise, Kommandos und Beispiele hier beziehen sich auf die Windows-Plattform.

4.3 Softwarewerkzeugkette
Eclipse und SVN

Man nehme pure Eclipse (>= Galileo, Indigo, Juno)

- \+ CDT (bzw. gleich die C/C++ Installationsvariante)
- \+ AVR Eclipse plug-in (http://avr-eclipse.sourceforge.net/updatesite)
- \+ Subclipse (Required) 1.6.5 (http://subclipse.tigris.org/update_1.6.x)

WinAVR, GCC und Libraries

WinAVR installieren aus z.B.

```
26.02.2010      28.840.282     WinAVR-20100110-install.exe
```

nach

```
C:\Programme\WinAVR
```

Für den reinen Eclipse- (+Automake-) Betrieb braucht man keine Pfadergänzung. Gestaltet man Projekte so, dass Alles auch von der Kommandozeile bzw. (batch-) automatisiert verwendet werden kann, ist der Pfad um zwei WinAVR-Verzeichnisse zu ergänzen:

```
PATH=C:\bat;C:\programme\util;C:\programme\jdk\bin;
       C:\programme\WinAVR\bin;C:\programme\WinAVR\utils\bin;
         C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;
           C:\Programme\TortoiseSVN\bin
```

Warnung zu Windows 7: Das bei (deutschem) Windows 7 im Explorer als C:\programme\ angezeigte Verzeichnis heißt gar nicht so — man vergleiche mit dem dir-listing. Eine heimtückische Falle.

On-Board-Programmierung

Die Programmierung erfolgt „on board“ über die ISP-Schnittstelle; vgl. u.A. Bild 22 auf Seite 26. Die kleine Baugruppe mySmartUSB ist ein bewährter und schneller Weg von einem PC via USB zur ISP-Schnittstelle.

Hat man am Entwicklungsrechner direkt (oder über ein simples USB-Gerät) eine serielle Schnittstelle, benötigt man für weAut_01 kein Programmiergerät, sei es mySmartUSB oder ein anderes. Wenn man mySmartUSB dennoch zur Verfügung haben möchte, gehe man so vor:
Man installiere aus

```
26.02.2011  4.462.916  tool_usb-treiber-myavr-board-v5.4.24.zip
25.10.2009  5.479.464  CP210x_VCP_Win2K_XP_S2K3.exe
```

für alles außer Windows 7 bzw. für Windows 7 aus

```
06.09.2011  4.463.927  tool_usb-treiber-myavr-board-v5.4.24-win7.zip
25.10.2009  5.492.504  CP210x_VCP_Win7.exe
```

oder neuere, passendere Dateien in den zwei Schritten Entpacken und Installieren:

```
1.)  jar xfv tool_usb-treiber-myavr-board-<vers>.zip
2.)  CP210x_VCP_<target>.exe
```

Erst nach (!) dem Lauf des Installationsprogramms den mySmartUSB anschließen.

Den verwendeten (neuen) Comm-Port kann man bei installiertem Java-Framework Frame4J klären mit

```
java ShowPorts
```

oder durch Nachschauen im Windows-Gerätemanager. Den neu hinzugekommenen Pseudo-Comm-Port (z.B. com3) muss man man in die (Eclipse-) Konfiguration, und ggf. in Batch-und make-files eintragen bzw. sinngemäß so verwenden.

```
avrdude -p atmega1284p -c avr911 -P com3 -U flash:w:leprog.hex
```

Hinweis: Man kann den verwendeten Pseudo-Comm-Port im Windows-Gerätemanager auf einen anderen freien Port ändern. Letzteres ist sinnvoll, wenn man dasselbe Entwicklungsprojekt bzw. make-File auf mehreren Arbeitsstationen verwenden will. Dann sollte der Einfachkeit halber überall der gleiche Port verwendet werden.

Hinweis 2: Häufig funktionieren bei avrdude „zweistellige“ Com-Ports, also z.B. COM15, nicht. Falls man von diesem Bug betroffen ist, muss man sich auf den Bereich COM1 .. COM9 beschränken. Gelegentlich hilft auch `-P \\.\COM15` [sic!] gegen die „handle“ Fehlermeldung anstelle des einfachen `-P COM15` .

Hinweis 3: Windows 7 reagiert auf das Ziehen einiger auch in Programiergeräten verbauter USB2serial converter trotz korrekt installierter Treiber mit Absturz. Es handelt sich um hoffentlich vorübergehende Probleme einiger Hersteller mit dem 64 bit-System.

Hinweis 4: Unter Linux sind für mySmartUSB[light] keine zusätzlichen Treiber erforderlich. -P /dev/ttyUSB0 verbindet zum Gerät --- mit dem dann die Programmiersoftware (zumindest direkt installierte Binaries) i.A. Nicht zurecht kommt. Hier nimmt man dann ohne Nachteile eine vorhandene serielle Schnittstelle direkt (s.o.).

5. Die Grundsoftware weAutSys

Im Rahmen der durch die Hardware gegebenen Randbedingungen (siehe letztes Kapitel) gibt es vielfältige Möglichkeiten des Einsatzes von Software-Eigenentwicklungen und von open source Paketen.

Empfohlen wird der Einsatz der Grundsoftware / des run time Systems weAutSys als Basis.

5.1 Lizenz

weAutSys ist eine von weinert – automation entwickelte Grundsoftware unter kommerzieller Lizenz. Ein Teil der Komponenten von weAutSys enthält auch modifizierte open source Software mit LGPL bzw. BSD Lizenz. Diese Teile bleiben open source und behalten ihre jeweilige LGPL bzw. BSD Lizenz.

LGPL und BSD beeinträchtigen *nicht* die kommerzielle Lizenz von Produkten, die so lizenzierte Teile modifiziert oder unverändert inkorporieren.

Die übrigen Teile und damit das Gesamtpaket weAutSys stehen unter kommerzieller Lizenz. Mit dem Erwerb einer hiermit ausgelieferten Baugruppe erwirbt man das Nutzungsrecht für die Software auf der selbigen Baugruppe. Jede andere Verwendung, das Rückübersetzen und re-engineering, und jede Verbreitung in binärer Form oder als Quelltext stellen einen strafbaren Verstoß gegen die Nutzungbedingungen dar.

Neben diesem mit eine erworbene Baugruppe gekoppelten Nutzungsrecht können weitere darüber hinausgehende Lizenzen erworben werden.

Softwarekomponenten

Komponente	Lizenz
gnu-C-libraries	LGPL
AVR-C-libraries	LGPL
Grundsystem	kommerziell (weinert - automation)
Systemtreiber	kommerziell (weinert - automation)
ProtoThreads	BSD - Lizenz
lwIP	BSD - Lizenz
ENC28J60-Treiber	kommerziell (weinert - automation)
SMC-Treiber	kommerziell (weinert – automation)
bootloader	open source (bzw. gleiche Lizenz wie AVR109 loader)

5.2 Parallelbearbeitung und Ablaufsteuerung

Der Anwendungsprogramierer setzt bei weAutSys seine Software in "vorgefertigte" Threads, im PLC / SPS-Sprachgebrauch auch "Zyklen genannt", ein. Diese werden entweder periodisch oder ereignisgesteuert aufgerufen. Es gibt u.A.

- 1ms-Zyklus
- 10ms-Zyklus
- 100ms-Zyklus
- 1s-Zyklus

- Anlauf
- Bedieneingabe
- Kommunikation

Die Periode der zyklischen Threads kann einfach durch Nutzung alle ungerade, alle gerade, alle n Mal beliebig ganzzahlig verlängert werden. Darüber hinaus stehen beliebig Timer, auch mit (sehr) langen Laufzeiten, in Millisekunden-Granularität zur Verfügung.

Die periodischen Nutzer-Zyklen können angehalten und wieder gestartet werden (PLC run und PLC stop).

5.3 Kommunikation und Utilities

weAutSys unterstützt die Kommunikation via RS.323 mit oder ohne symmetrische CTS/RST-Flusskontrolle. Mit letzterer sind P2P-Verbindungen zwischen zwei Baugruppen einfach möglich.

weAutSys unterstützt die Ethernet-Kommunikation und bietet alle Möglichkeiten des TCP/IP-Stack uIP. DHCP und im EEPROM hinterlegte Konfigurationen werden unterstützt.

Darüber hinaus bietet weAutSys zahlreiche (Hilfs-) Funktionen zu Arithmetik, Kommunikation, Formatter, Parser und I/O, welche in Hinblick auch die Atmel 8bit RISC-Prozessoren optimiert sind.

Darüber hinaus sind einige Anwendungsprotokolle, wie NTP, Modbus, Telnet, unterstützt bzw. implemetiert.

5.4 Dokumentation

Zu weAutSys gibt es Veröffentlichungen und eine separate Dokumentation in englischer Sprache. und die Dokumentation des C-Codes, erzeugt aus den Quellen mit Doxygen, siehe auch

https://weinert-automation.de/files/weAutSys/doxygen/ .

Anhang
I Inbetriebnahmeschritte

Sichtkontrolle

Alles bestückt

Visueller Vergleich mit Bild 27, z.B.:

Quarz am µController 20MHz und am Ethernet-Chip 25 MHz

Die vier "großen" Puffer-Elkos haben 330µF und eine Spannungsfestigkeit von 35V

Keine offensichtlichen Kratzer, Lötspritzer, Kurzschlüsse

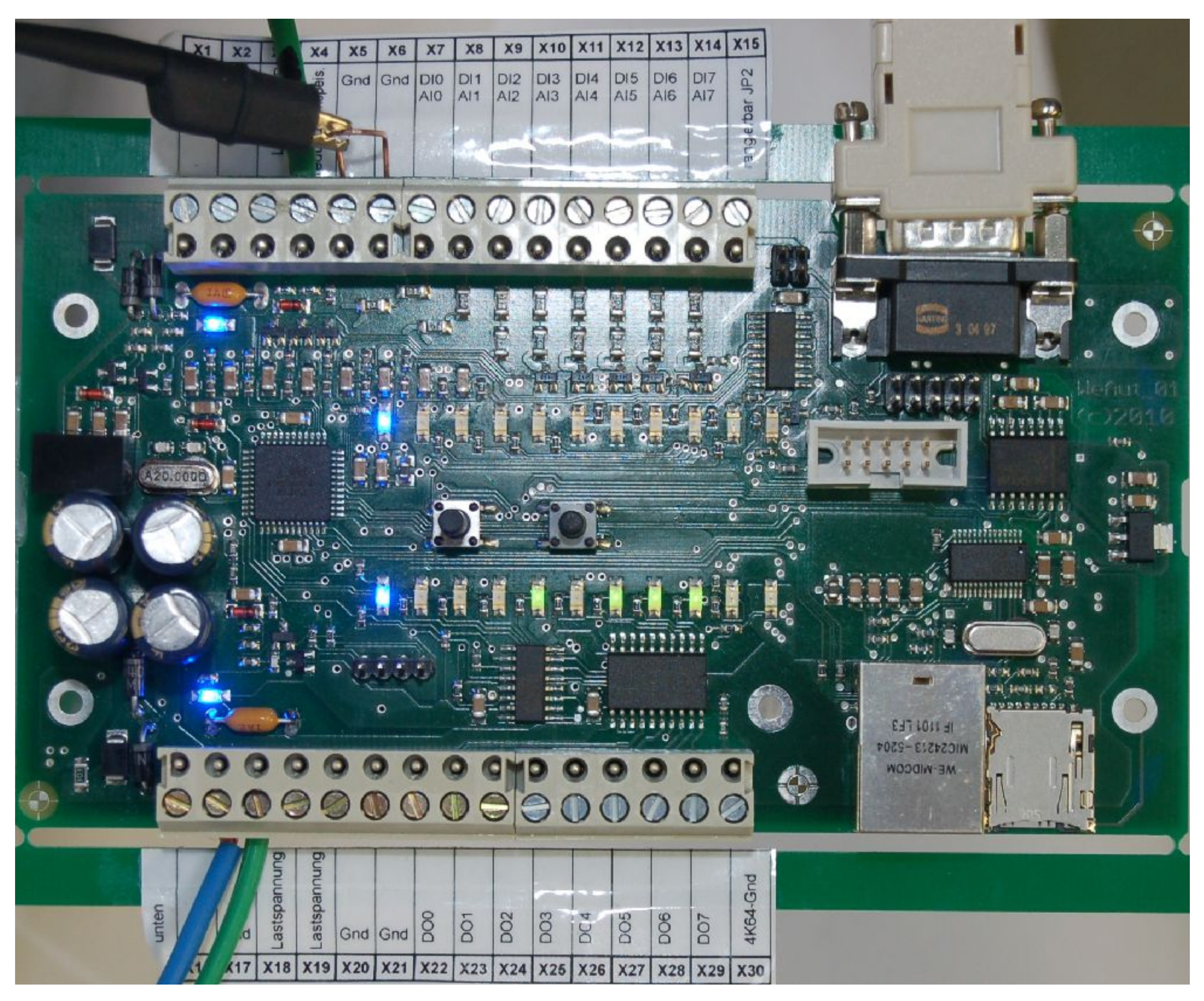


Bild 27: Versuchsaufbau für die Erstinbetriebnahme

Für die nächsten Schritte benötigt:

Netzgerät 0..35V, 1A mit Spannungs- und Strommessung; Schutzwiderstand 10 Ohm
Spannungsmesser

Hochfahren der redundanten Einspeisung

0..30V mit Vorwiderstand (etwa 10 .. 15 Ohm) und Strommessung

Stopp über 300mA (Fehlbestückung, verpolte Bauelemente)

Kontrolle der Vcc (5V) und der +3V3 (3,3V)

Stopp, wenn hängt oder +10%

Aufzeichnen der Stromaufnahme (muss oberhalb 10V Einspeisung fallen)

Kontrolle des Einsatzes der (oberen) Leuchtdioden (ab ca. 10V an, ab 12V hell)

Ermitteln des Einsatzes der Transzorp-Diode (muss deutlich über 24V und deutlich unter 35V sein)

Testen negative Einspeisung (keine Stomaufnahme, kein "Durchschlagen" zum Wandlereingang, messbar an Klemme X3)



Bild 28: Messpunkte
für Vcc (5V +/- 0,1V) 2 Durchkontaktierungen
und für 3V3 (3,3V +/- 0,1V) "Pad" des Reglers

Hochfahren der Lastspannung

0..30V mit Vorwiderstand (etwa 10 .. 15 Ohm) und Strommessung

Stopp über 300mA (Fehlbestückung, verpolte Bauelemente)

Kontrolle der Vcc (5V) und der +3V3 (3,3V)

Stopp wenn hängt oder +10%

Aufzeichnen der Stromaufnahme (muss oberhalb 10V Einspeisung fallen)

Kontrolle des Einsatzes der (unteren) Leuchtdioden (ab ca. 10V an, ab 12V hell)

Ermitteln des Einsatzes der Transzorpdiode (muss deutlich über 24V und deutlich unter 35V sein)

Testen negative Einspeisung mit Vorsicht:

deutliche Stomaufnahme ab 0,5V ist OK bzw. muss so sein (Verpolschutz),
Lastspannung darf nicht unter -0,9V kommen
sonst Stopp (Fehlbestückung)

Man darf hierbei nicht über 1A gehen, sonst ist die Sicherung F1 gefährdet.

Ansprechen des Prozessors, Setzen der Fuses

ISP an Programmiergerät bzw. direkt seriell anschließen, vgl. die Bilder 14 bis 20 ab Seite 16. Hier ist es i.a. empfehlenswert, zunächst "seriell direkt" (pony, bit bang) zu nehmen, da "mySmartUSB" fuses nicht schreiben kann. (mySmartUSB light kann es.)

Auslesen der fuses gemäß Listing 29,das auch den erwarteten Lieferzustand eines fabrikneuen ATmega1284 zeigt.

```
$$ entweder ser $> avrdude -p atmega1284p -c ponyWeAut -P com1 -v
$$ oder via USB $> avrdude -p atmega1284p -P com7 -c avr911  -n -v

avrdude: Device signature = 0x1e9705
avrdude: safemode: lfuse reads as 62
avrdude: safemode: hfuse reads as 99
avrdude: safemode: efuse reads as FFs.c
```

Listing 29: .: Auslesen der fuses mit seriell direkt (bit bang) oder mySmartUSB o.ä.

Setzen der Fuses auf `lfuse = FF, hfuse = D9` und `efuse = FC` mit

```
  avrdude -p atmega1284p   -c ponyWeAut -Pcom1  -U lfuse:w:0xFF:m
..avrdude -p atmega1284p   -c ponyWeAut -Pcom1  -U hfuse:w:0xD1:m
  avrdude -p atmega1284p   -c ponyWeAut -Pcom1  -U efuse:w:0xFC:m
```

Hinweis: Die wesentlichen Unterschiede zum Lieferzustand sind JTAG aus, BOD an, low power crystal, f*1, long start-up.

Kontrolle des Ergebnisses durch Auslesen

Optional Kontrolle ob Quarz mit 20MHz (50ns Periode) schwingt. Tastkopf mit Vorwiderstand verwenden (1:10), sonst "bläst" man den Oszillator häufig aus.

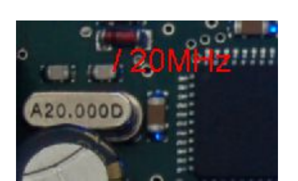


Bild 30: Messpunkt: rechts am "rechten Kondensator" C17

Falls (im übernächsten Schritt) die Testsoftware und insbesondere die serielle Kommunikation fehlerfrei läuft, kann die Kontrolle der Quarzgeneratoren durch das Messen der korrekten Quarzfrequenz mit Messgeräten (gemäß Bild 30) entfallen. Die serielle Kommunikation ist recht empfindlich gegenüber Abweichungen von der korrekten Taktfrequenz.

Hinweis: Falls der Quarz nicht schwingt, versuche man es mit lfuse = F7.
Das ist "full swing" statt "low power" (d.h. kaum messbar höherer Stromaufnahme).
Also sinngemäß:

```
  avrdude -p atmega1284p   -c ponyWeAut -Pcom1  -U lfuse:w:0xF7:m
```

Wenn es nun geht, ist auch alles in Ordnung. Im Rahmen der Exemplarstreuung von Wandler (5V), Prozessor und Quarz gibt es Kombinationen, in denen die Einstellung "low power" nicht sicher schwingt. Außer in sehr stromverbrauchkritischen Anwendungen mag man sich auch von vornherein für die "full swing" Einstellung entschließen.

Hinweis 2: Das Setzen der fuses ist kritisch. Man kann damit den µController und so die Baugruppe unbrauchbar machen; Reparaturhinweis ggf. weiter weiter unten.

Bei der Gelegenheit (und ggf. auch als Kontrolle der Mess-einrichtung) gleich den Quarzoszillator des 28J60 an C14 kontrollieren (25 MHz, 40 ns).

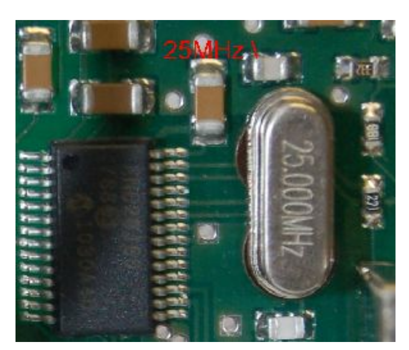


Bild 31: Messpunkt am Kondensator neben dem Quarz

fuse	Lieferzust.	WeAut std	alternativ	Änderung std.>alternativ
lfuse	62	FF	F7	Quarz schwingt sicherer und schneller an
hfuse	99	D1 / D0	D9 / D8	EEPROM wird mit chip erase gelöscht
efuse	FF	FC		

Tabelle 32: Übersicht fuse settings

Meist ist es gewünscht, dass der Inhalt des EEPROMS eine Neuprogrammierung „über-lebt“, z.B. wenn dort Konfigurationsdaten, wie die MAC-Adresse, hinterlegt sind. Anderenfalls setze man die hfuse statt auf D1/D0 auf D9/D8 mit sinngemäß

```
..avrdude -p atmega1284p   -c ponyWeAut -Pcom1  -U hfuse:w:0xD9:m
```

Falls vorhanden kann man nun die ISP auf ein USB-Programmiergerät "mySmartUSB" umsetzen, um die serielle Schnittstelle freizumachen.
In diesem Falle erneutes Probeauslesen der gerade gemachten fuse settings nun mit:

```
  avrdude -p atmega1284p -P com7 -c avr911  -n -v
```

Erinnerung: Fuses lesen kann der mySmartUSB, nur schreiben kann er sie nicht.

Hinweis: Die hfuse-Einstellungen D0 oder D8 anstatt D1, D9 gelten für die (empfohlene Verwendung der weAutSys bootloaders.

Einschub: fuse repair

Falls man aus Unachtsamkeit die fuse Einstellungen "zerschossen" hat, kann dass leider so weit gehen, dass der Prozessor mit der ISP-Schnittstelle nicht mehr "spricht".
Dann kann man auf diesem Wege die fuses auch nicht mehr korrekt setzen.

Aus dieser Sackgasse helfen Programmiergeräte mit anderen "Nicht-ISP-" Ansätzen (JTAG oder 12V Programmierspannzng).

Oft geht es aber auch viel einfacher mit dem schlichten Einspeisen eines externen Prozessortakts: Rechteck, TTL oder CMOS, etwa 1 bis 2 MHz,
am besten mit einem 50 Ohm Entkopplungswiderstand.

Einspeisepunkt wäre "die linke Seite des linken Kondensators", also im Bild 30 gespiegelt zum Messpunkt.

Wenn der µC nun mit der ISP-Schnittstelle spricht, ist man natürlich gerettet.

Meilenstein: µC läuft

Wenn man hierher gelangt ist

- läuft der Prozessor in der richtigen Grundbetriebsart (Frequenz etc.) und
- "spricht" mit Programmiergeräten (pony und mySmartUSB) via ISP

Falls das nicht geht ...
... nicht weiter, sondern klären / reparieren!

Testsoftware aufspielen

Mit der Testsoftware (weAutSys + Demo-/Testprogramm) und den entsprechenden make-Einstellungen generiert man die Software mit:

```
$$$>make clean
$$$>make all
```

Geladen wird sie mit einem Programmiergerät "smartUSB"

```
$$$> avrdude -p atmega1284p   -P com7   -c avr911   -U flash:w:main.hex
```

oder mit einem Programmiergerät smartUSB light"

```
$$$> avrdude -p atmega1284p   -c avrisp2  -P com4 -U flash:w:main.hex
```

oder via "seriell direkt"

```
$$$> avrdude -p atmega1284p   -P com1   -c ponyWeAut -U flash:w:main.hex
```

oder via serielle Schnittstelle und dem weAutSys bootloader

```
$$$> avrdude -p atmega1284p -c avr109 -b 38400 -P com1 -U flash:w:main.hex
```

Nach Einspielen der (auf weAutSys beruhenden) Test- Demosoftware:

Serielle Kommunikation — z.B. mit HTerm 0.8beta — testen. Einstellung:

```
38400 Baud, 8 Bit, 1 Stop, no parity, no flow control
newline at LF, send on enter LF, [connect]
```

Terminaleingabe "abort" muss zu Abbruch und Neustart mit "watchdog" als "reset cause" führen.

Reset-Taste muss zu Abbruch und Neustart mit "ext. reset" als "reset cause" führen.

Eine diesbezügliche Anzeige erhält man mit dem Kommando "runInfo".

Anschließen der Ethernet-Schnittstelle (direkt oder indirekt) an einen DHCP-Server. Der Server oder der nächstgelegene Switch muss 10MBit/s können.

Kommando "IPconfig" an der seriellen Schnittstelle muss u.a. die erhaltene IP-Adresse zeigen. Wenn dieser Test funktioniert, ist auch der 25 MHz-Takt des ENC28J60 OK und muss nicht eigens geprüft werden.

A Abkürzungen

AC Wechselstrom; alternating current

AI.. analogue input / Analogeingabe

ANSI American National Standards Institute

AO.. analogue output / Analogeingabe

API Application Programme Interface

ARP Address Resolution Protocol

bl blau

B&B Bedienen und Beobachten (von Prozessen)

BIST Built-in Self-Test

BSD Berkeley Software Distribution (Lizenz)

CRC Cyclic Redundancy Check:; Prüfkodeverfahren auf Polynomdivision beruhend

C/S Client-Server

/CS.. chip select (i.a. 0-aktiv, signalisiert durch /)

CSMA/CD Carrier Sense Multiple Access with Collision Detect

DHCP Dynamic Host Configuration Protocol

DI.. digital input / Binäreingabe, i.A. 24V

DNS Domain Name System

DO.. digital output / Binärausgabe, i.A. 24V

EAI Electronic Industries Alliance

EAI323 neuere Bezeichnung für RS232/V.24

ED Einschaltdauer (relative Angabe in % der Zeit AN)

EMC, EMV Elektromagnetische Verträglichkeit, Störsicherheit, Störfestigkeit

EMI Elektromagnetische Interferenz / Störung

EUI-48 48-bit Extended Unique Identifier: EUI-48™ is a concatenation of a 24-bit OUI value assigned by the IEEE-RA and a 24-bit extension identifier assigned by the organisation with that OUI assignment; ersetzt MAC-48; (es gibt auch EUI-64;...)

FAQ Frequently Asked Questions (Hilfetexte in Frage-Antwort-Form)

ge, gn gelb, grün

Gnd ground, Erde, Bezugspotential, 0V

GNU GNU is not Unix

GPL [GNU] General Public License (infizierende open source Lizenz)

GSS Generic Security Service

GUID Globally Unique Identifier

Hi High, hoher Pegel, VCC, 5V/3,3V, Lastspannung, i.A. logische 1, true

HTML Hypertext Markup Language [RFC 1866]

HTTP Hypertext Transfer Protokoll.
Internet-Protokoll zur Übertragung von Seiten.

HTTPS HTTP über SSL. Abgesicherte Übertragung.

HW Hardware

IC integrated circuit; integrierter Schaltkreis

ICMP Internet Control Message Protocol

IEC International Electrotechnical Commission

IEEE Institute of Electrical and Electronics Engineers

IEEE-RA IEEE Registration Authority

ISO International Standardization Organization

ITI Information Technology Industry Council

IP Internet Protocol

ISP in system programming

J2ME Java 2 Micro Edition

JAR Java Archive. (.zip + Semantik)

JDK Java Development Kit; der Werkzeugsatz für die Entwicklung mit Java

JRE Java Runtime Environment; JDK-Subset ohne Entwicklungswerkzeuge.

JTAG Joint Test Action Group

LAN Local area network; Datennetz für mittlere Entfernungen

LED Leuchtdiode

LGPL [GNU] Lesser General Public License bzw. früher Library General Public License

Lo Low, niedriger Pegel, Gnd, 0V, i.A. logische 0, false

LV load voltage; Lastspannung, Lastversorgung, i.A. nominell 24V, gel. auch 12 V

MAC Media access control

MEVA Labor für Medien und verteilte Anwendungen (meva-lab.de)

MISO SPI-Signal Master in / slave out

MOSI SPI-Signal Master out / slave in

MS Microsoft

µC µ-C Mikro-Controller, µController

NAMUR Normenarbeitsgemeinschaft für Mess- und Regeltechnik in der Chemischen Industrie

NIC network interface card / circuit

NT Betriebssystem Windows NT (MS)

OMG Object Management Group

OS.. operating system / Betriebssystem

OUI Organizationally Unique Identifier; 28 Bit; z.B. genutzt als erste drei (von 6) Bytes einer MAC-Adresse (MAC-48; EUI-48), OUI wird verteilt / verkauft vom IEEE-RA

P2P point to point

PAM Pluggable Authentication Module

PA, PB, PC, PD Port A..D

PC Personal Computer

PCB Printed circuit board (Leiterplatte)
process control block (process / thread Datenstruktur, Betriebssystem)
protocol control block (Verbindungsdatenstruktur bei lwIP)

PE protecting earth, Schutzerde

PLC programmable logic controller (deutsch SPS)

PWM Pulsweitenmodulation
(Darstellung von 0..100% als relative An-Zeit eines digitalen Ausgangs)

rt rot

R&D Research and Development

RARP Reverse Address Resolution Protocol

RDF Resource Description Framework (W3C)

RISC Reduced instruction set computer

RPC Remote Procedure Call

RX Receive, Empfang

sw schwarz

SCK SPI-Signal shift clock

SMC Small memory card, kleine Speicherkarte bzw. Steckplatz / Slot dafür

SMTP Simple Mail Transfer Protocol

SNTP Simple Network Time Protocol

SPI serial periphery interface

SPS speicherprogrammierbare Steuerung (englisch PLC); kleines Automatisierungsgerät

SQL Structured query language, Datenbankbearbeitungssprache

SSD Start-of-Stream delimiter:

SSL Secure Socket Layer. Protokollschicht zu Absicherung.

SSPI Security Support Provider Interface

SVN Subversion

TCP Transmission Control Protocol

TM Trade Mark (Warenzeichen)

TWI Two Wire Interface (entspr. Philips® Inter-IC bzw. I2C bus)

TX Transmit.; Senden

UDP User Datagram Protocol

UL Lastspannung; siehe LV

UML Unified Modelling Language

URI Uniform Resource Locator

UTP Unshielded Twisted Pair wire

Vcc Voltage Collector Collector; positive Versorgungsspannung (i.A. 5V oder 3,3V)

Vdd Voltage Drain Drain; positive Versorgungsspannung (i.A. 3,3V oder 5V)

VTG target voltage (i.A. via ISP; =VCC)

V, Vss, Veff Volt, ss: Spitze-Spitze / peak to peak, eff: effektiv (quadratischer Mittelwert)

W2K8 Betriebssystem Windows Server 2008 (MS)

W3 Amerikanische Kurzform für WWW

W3C World Wide Web Consortium

WeAut Kurzform von weinert - automation (weinert-automation.de)

WebDAV Web-based Distributed Authoring and Versioning

WS Workstation

WSDL Web Services Description Language

XML eXtensible Markup Language

L Literatur

[1] Atmel Corporation,
8-bit Microcontroller with 16/32/64/128K Bytes In-System Programmable Flash
Atmega164A Atmega164PA Atmega324A Atmega324PA
Atmega644A Atmega644PA Atmega1284 Atmega1284P — Summary
(ausführlicher Überblick, atmel-Dokument 82725.pdf)

[2] Atmel Corporation,
8-bit Microcontroller with 16/32/64K Bytes In-System Programmable Flash
Atmega164P Atmega324P ATmega644P — Automotive
(umfassendes Datenblatt, atmel-Dokument doc7674.pdf = db_atmega164p-324p-644p.pdf)

[3] Atmel Corporation, 8-bit AVR Microcontrollers Application Note
AVR040: EMC Design Considerations (atmel-Dokument doc1619.pdf)

[4] Atmel Corporation, 8-bit AVR Microcontrollers Application Note
AVR042: AVR Hardware Design Considerations (atmel-Dokument doc2521.pdf)

[5] Atmel Corporation, 8-bit AVR Microcontrollers Application Note
AVR109: AVR Self programming (atmel-Dokument doc1644.pdf)

[6] Atmel Corporation, 8-bit AVR Microcontrollers Application Note
AVR1308: Using the XMEGA TWI (atmel-Dokument doc8054.pdf)

[7] Atmel Corporation, 32-bit AVR Microcontrollers Application Note AVR32817:
Getting Started with the 32-bit AVR UC3 Software Framework lwIP TCP/IP Stack
(atmel-Dokument doc32817.pdf)

[8] Atmel Corporation, 32-bit AVR Microcontrollers Application Note AVR32733:
Placing data and the heap in external SDRAM (atmel-Dokument doc32121.pdf)

[9] I2C-bus specification and user manual Rev. 03 — 19 June 2007
User manual: UM10204.pdf

[10] Allegro MicroSystems, Inc.
UDN2987x-6 DABIC-5 8-Channel Source Driver with Overcurrent Protection 2987-6.pdf

[11] Microchip Technology Inc. ENC28J60 Data Sheet
Stand-Alone Ethernet Controller with SPI Interface (Dokument 39662c.pdf)

[12] Microchip Technology Inc., M. Simmons. Application Note AN 1120
Ethernet — Theory of Operation (Dokument 01120a.pdf)

[15] NXP Semiconductors Product Data Sheet
BCV27; BCV47 NPN Darlington transistors (Dokument BCV27_BCV47.pdf)

[20] William von Hagen
The definite guide to GCC second edition, apress 2006

[22] Sandra Loosemore, Richard M. Stallman, Roland McGrath,
Andrew Oram, Ulrich Drepper The GNU C Library Reference Manual
Edition 0.12 last updated 2007-10-27 for version 2.8 (libc.pdf)

[23] Adam Dunkels, SICS (Swedish Institute of Computer Science)
The uIP Embedded TCP/IP Stack The uIP 1.0 Reference Manual
June 2006 (uip-refman.pdf)

[24] Adam Dunkels, SICS (Swedish Institute of Computer Science)
Full TCP/IP for 8-Bit Architectures 2003 (mobisys2003.pdf)

[25] Adam Dunkels, SICS (Swedish Institute of Computer Science)
Design and Implementation of the lwIP TCP/IP Stack 2001 (lwip.pdf)

[26] Adam Dunkels, SICS (Swedish Institute of Computer Science) et alii
Protothreads: Simplifying Event-Driven Programming of Memory-Constrained Embedded Systems 2006 (dunkels06protothreads.pdf)

[27] Sathyanarayanan Thammanur and Chris Borrelli (Xinilx)
TCP/IP on Virtex-II Pro Devices Using lwIP
XAPP663 (v1.1.1) August 30, 2004, (xapp663.pdf)

[41] IEEE Standards Organisation
Guidelines for use of the 24-bit Organizationally Unique Identifiers (OUI) (eui.pdf)

[42] IEEE Standards Organisation
Guidelines for use of a 48-bit Extended Unique Identifier (EUI-48™) (eui48.pdf)

[43] Richard M. Stallman, et al.
GNU Coding Standards (standards.pdf; updated January 27, 2011)

[44] Richard M. Stallman, Roland McGrath, Paul D. Smith
GNU Make A Program for Directing Recompilation (Version 3.82 July 2010 make.pdf)

[45] ANSI International Standard ISO/IEC 9899:1999 (E)
Programming languages — C (Second edition1999-12-01 ansi_c.pdf)

[51] Maxim Application Note 3969 (AN3969.pdf)
SD Media Format Expands the MAXQ2000's Space for Nonvolatile Data Storage

[52] ...

[53] SD Group (Panasonic Corporation, SanDisk Corporation, Toshiba Corporation)
Technical Committee SD Card Association
SD Specifications, Part 1, Physical Layer, Simplified Specification Version 3.01
(May 18, 2010; Part_1_Physical_Layer_Simplified_Specification_Ver3.01_Final_100518.pdf)